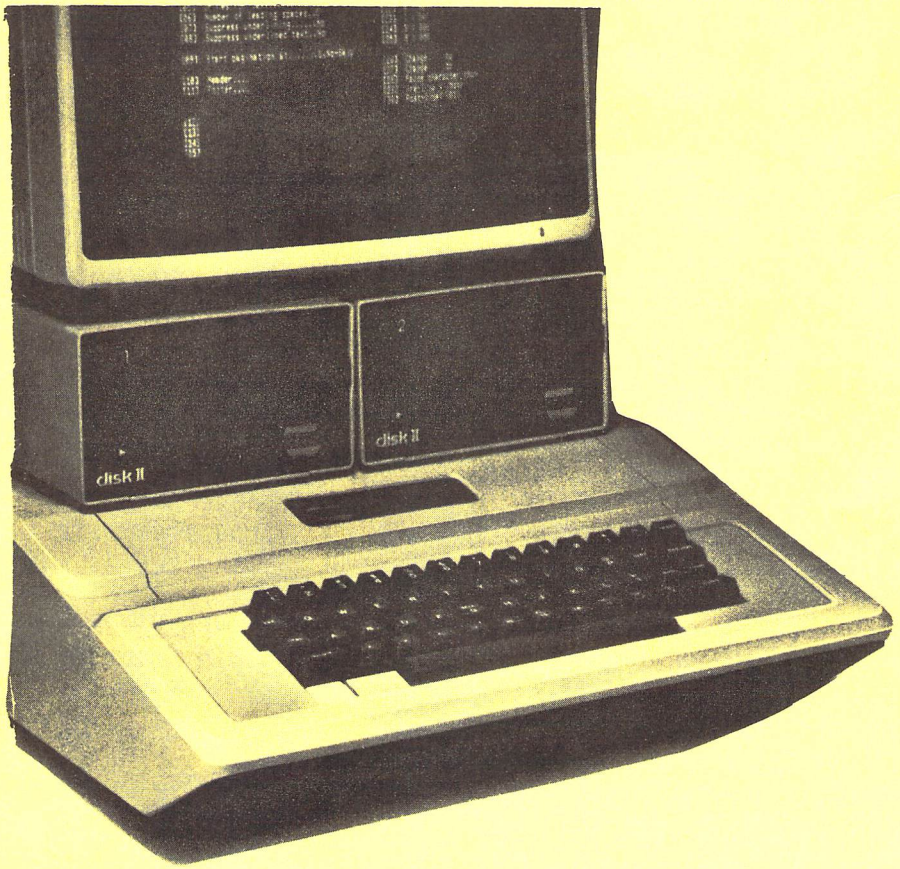
THE SCARLETT LETTER

Vol 2 No 6 July 1983

*** TABLE OF CONTENTS ***



THE BIG RED APPLE CLUB

*** MASTHEAD ***

THE SCARLETT LETTER is published monthly by THE BIG RED APPLE CLUB, 1301 N 19th, Norfolk, Nebraska 68701. Phone (402) 379-3531. Editor: John Wrenholt.

ADVERTISING RATES: 1/4 page $25.00, 1/2 page $50.00, full page $100.00. All ads submitted must be camera ready for publication. Prepayment required for all first time advertisers.

CHANGES OF ADDRESS should be sent to BRAC, 1301 N 19th, Norfolk, NE 68701. Mailings are by bulk presorted third class mail except for foreign or servicemen's addresses.

SUBMITTING ARTICLES: Handwritten articles are acceptable but articles on disk are preferred. All disks will be returned by first class mail. Applewriter compatible files are preferred but other word processors are acceptable. If your article includes a program listing please submit it on disk. If your article is published you will receive coupons good for free Disks of the Month or Disk.Network issues.

MEMBERSHIP is open to all. Dues are $12.00 a year. Add $16.00 for overseas airmail postage. Please make payment in U.S. dollars. Membership benefits include a subscription to the "Scarlett Letter", a sample isssue of Disk.Network, and full rights to use the Software Library. Each new member will receive a complete listing of all available programs in the library.

SOFTWARE LIBRARY

The "BIG RED APPLE CLUB" has a large library of Public Domain Apple software from which members can obtain copies. This provides an inexpensive and easy way to add programs to your personal software library.

When ordering state the disk sides you want copied by number. Requests for individual programs will not be honored. All disks are DOS 3.3 unless otherwise stated. Each member furnishs blank diskettes for all requested copies. If desired, blank diskettes are available from B.R.A.C. for $2.50. (Nebraska residents add 3 1/2% sales tax).

To save on postage, programs can be copied onto both sides of your diskettes. To be able to use the back side of a disk you just use a paper punch to make a notch on the disk directly opposite the existing notch.

There is a $1.00 charge for all copies made from the software library. This fee will help cover the costs of maintaining and upgrading the library. Postage will be paid by B.R.A.C. for addresses within the United States.

All programs in the library are believed to be in the public domain. If you find a program in the library which you feel shouldn't be there, please let us now so we can remove it. Because these programs are public domain, all programs are copied as is and no guarantee is made that they work.

APPLE is a registered trademark of Apple Computer Inc.

WHAT IS PUBLIC DOMAIN SOFTWARE?

Someone asked me to define what is meant by 'public domain' software. Defining 'public domain' software is an easy task. Public domain simply means that the program's original author has freely given his/her program to the general computing public do with as they see fit. A program which is truly public domain has no strings attached. You can use it or distribute it in any manner you choose. Sometimes authors release a program to the public, but add the stipulation that it can only be distributed for non-commercial purposes.

Most programs which qualify as public domain have been written by members of Apple user groups such as BRAC. A member writes a program of which he is proud of and then donates it to the club's software library so that it can be shared with other members of the club. In order to increase the size of their library, clubs trade their programs back and forth.

Programs which are published in magazines are generally NOT public domain. Even though you subscribe to the magazine and have gone to all the sweat to type in the program listings, this does not give you the right to give even one friend a copy. Here's what Wayne Green, noted editor of several computer magazines including 'inCider' has to say on the subject. "Another totally forbidden practice is to make copies of inCider articles for cheapskates. Let the bums buy their own copies. And the hell we have reserved for the lowlife who keys in a program and then lets a friend copy it can't be put into print." (inCider Magazine, Aug 1983, page 6). Mike Harvey, editor of 'Nibble' has also expressed similar feelings although in a somewhat nicer tone.

Some magazines, most notably 'Softalk' and 'Softline', have placed certain programs in the public domain. Two examples of this are BASICALC and SOFTGRAPH both of which were originally published in Softalk. These, however, are exceptions to the rule.

When we receive new programs for inclusion into the library, we try to determine their origin. If they were originally published in a magazine, it is our policy not to include them in the library. Since we do not subscribe to all the computing magazines, determining the source of a program can be very difficult. It is therefore a good idea to always include the original source of a program in REM statements.

What if a program is copyrighted? Can it still be public domain? In order to be sure, that we are clearly on legal ground, it is the club's policy not to distribute programs which are copyright unless permission to distribute the program as public domain is granted by the copyright owner.

Hopefully this will give you a clear idea of what public domain software is. Unless stated otherwise, all programs which are publiished in the "Scarlett Letter" and on "Disk.Network" are public domain.

*** ***

THE WOZ IS BACK!

Steve Wozniak designer of the Apple][has returned to the company he cofounded seven years ago.

Wozniak had not worked for Apple since he was injured in a plane crash in February 1981. Although, it's not clear exactly what position Woz will fill at Apple, he has stated that he will probably be working on new projects for the Apple][and ///.

Wozniak has not been idle for the last few years. During his absence from Apple, he completed his undergraduate college degree at the University of California at Berkley, starred in a Datsun TV commercial, and sponsored two US music festivals in southern California.

NOS BASICODE - PART 3

by Jack Decker
Reprinted from Northern Bytes
Published by Microcomputer Users International

This is the final installment of the series on NOS BASICODE, a universal cassette tape format. In this month's installment we shall provide the source code for the read and write routines and the instructions for using them. (Editor's note: Save yourself some typing. Both the source and object code for NOS BASICODE is provided on the third issue of Disk.Network.)

Using the translation program developed for the Apple computer, you should be able to read and write according to the BASICODE standard. The total system consists of:

- an Apple II or IIe computer with Applesoft in ROM
- a normal cassette recorder
- the translation program

The translation program consists of two parts: the writing routine called WRITE.BC and the reading routine called READ.BC. Both of the routines make use of the standard cassette which is normally used by the Apple, so apart from this, and the translation program, nothing else is needed. Using the writing program it is possible to record an Applesoft program onto cassette using the BASICODE standard. The reading routine takes a program in BASICODE and stores it as an Applesoft program. After a few changes (adaptation of BASICdialect) the program can be 'saved' or 'runned'.

The translation program will work with, or without, DOS (Diskette Operating System). The memory capacity of your Apple will determine the length of programs that can be handled. The minimum must be 16K however.

No special interface is necessary. Only programs in Applesoft BASIC can be sent to other users. Machine language routines which are not in the program as pokes or data statements are not useable in BASICODE. If you want your programs to work on other brands of computers, then observe the BASICODE protocol carefully (e.g. maximum line length, build up of subroutines, etc.).

In order to load a program in BASICODE use the following procedure:

1. Connect the Cassette In port of your Apple to the tape recorder's earphone jack.

2. Bload 'READ.BC'.

3. Type in 'NEW' (return).

4. Find the beginning of the program to be loaded on the cassette recorder.

5. Type in 'CALL 960', but DO NOT press return yet.

6. Start the tape, and when you hear the header tone (a high-pitched squeal), then press return.

The program will now be read, and you will note flashing characters at the top right hand corner of the screen. This is in fact a display of the characters as they are being read. The loaded characters are set in a table at the same time, and these can be read later using Applesoft, as though you'd typed the program yourself.

If there is no loading error, then after the trailer has been heard, the listing will appear on the screen. The table is automatically converted to Applesoft. After this procedure, the program can be handled according to the user's wishes.

If the Apple does not detect the end of the tape, i.e. the trailer, you can break in yourself by pressing the 'RESET' key. If this is the case, then one or more errors have been read. To check where these are, type in CALL 965 (return). You will now see what has been read start to move across the screen. You can stop it by pressing the 'space'

key. If it appears that a number of errors have appeared, you can still transfer to Applesoft BASIC using the command CALL 970 (return). After this we can list the program from Applesoft and correct the errors.

Note that once the table has been read once in BASIC, it is no longer necessary, and in order to conserve memory space, destroyed. The commands CALL 965 and 970 are now rendered useless and therefore should not be used further.

In order to transmit a program in BASICODE, you need to:

1. Hook the Cassette Out port to the cassette player microphone jack.

2. Bload 'WRITE.BC'.

3. Load the Applesoft program you wish to send in BASICODE.

4. Type in on one line: CALL 960 : LIST : CALL 970 (don't press return yet).

5. Start the tape recorder in the record mode, and press return.

Using CALL 960 : LIST will send the listing of the program into a table, and this is recorded onto the tape using the BASICODE tones using CALL 970. Because the table takes some time to make, there will be a slight pause before the recording begins. The Apple will produce a pip to indicate the beginning and end of the program. After the program has finished, the cursor will return and the tape can then be stopped.

It is also possible to record only a part of a program, e.g. a subroutine. If the subroutine runs, e.g. between line numbers 150 and 310 inclusive, then type CALL 960 : LIST 150,310 : CALL 970. After touching the return key, the section desired will be recorded. In this way it is possible to make up a subroutine collection, which can be simply loaded into other programs using the BASICODE reading routine.

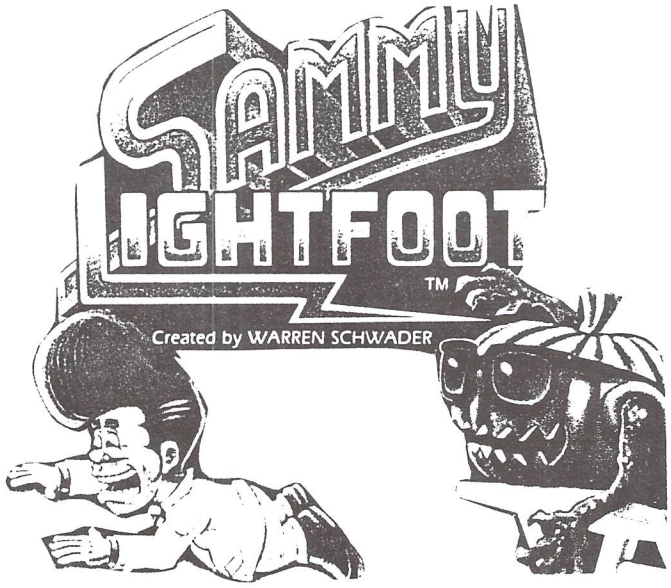
```
                        1010 *-------------------------------
                        1020 * BASICODE WRITE ROUTINE
                        1030 *    FOR THE APPLE ][
                        1040 *
                        1050 *
                        1060 * VERSION  SEPT 27, 1981
                        1070 *
                        1080 *-------------------------------
                        1090 *
                        1100 * by J.Hermann Peojhm
                        1110 *
                        1120 *-------------------------------
                        1130        .OR $280
0036-                   1140 CSWL   .EQ $36
0037-                   1150 CSWH   .EQ $37
0008-                   1160 CSOM   .EQ $8
0019-                   1170 PO1L   .EQ $19
001A-                   1180 PO1H   .EQ $1A
001B-                   1190 PO2L   .EQ $1B
001C-                   1200 PO2H   .EQ $1C
001D-                   1210 STOX   .EQ $1D
FF2D-                   1220 PERR   .EQ $FF2D
FBDD-                   1230 PIEP   .EQ $FBDD
00FF-                  1240 FLAG   .EQ $FF
001E-                   1250 PO3L   .EQ $1E
001F-                   1260 PO3H   .EQ $1F
0067-                   1270 STBL   .EQ $67
0068-                   1280 STBH   .EQ $68
                        1290 *-------------------------------
                        1300 * *****  CALL 960  *****
                        1310 *
                        1320 * INITIALIZE THE POINTERS
                        1330 * MOVE OUTPUT VECTORS CSW IN ROUTINE TABR
                        1340 *
                        1350 *-------------------------------
0280- A5 36             1360 INIT   LDA CSWL
0282- 85 06             1370        STA $6
0284- A5 37             1380        LDA CSWH
0286- 85 07             1390        STA $7
0288- A9 A7             1400        LDA #OUTL
028A- 85 36             1410        STA CSWL
028C- A9 02             1420        LDA /OUTL     NEW
028E- 85 37             1430        STA CSWH      OUTPUT VECTOR
0290- A9 00             1440        LDA #0
0292- 85 08             1450        STA CSOM      CHECKSUM = 0
0294- 85 FF             1460        STA FLAG      FLAG = 0
0296- A5 69             1470        LDA $69
0298- 85 19             1480        STA PO1L      MOVE THE POINTERS
029A- 85 1B             1490        STA PO2L      ON TOP BASIC
029C- A5 6A             1500        LDA $6A       PO1:BEGIN TABLE
029E- 85 1A             1510        STA PO1H      PO2:POINTER
02A0- 85 1C             1520        STA PO2H
02A2- A9 82             1530        LDA #$82       START OF TEXT
02A4- 4C A7 02          1540        JMP TABR      MOVE IN TABLE
                        1550 *-------------------------------
                        1560 *
                        1570 * THIS ROUTINE WILL GO THROUGH
                        1580 * BASIC AND MOVE THE LISTING FROM THE
                        1590 * PROGRAM INTO A TABLE.
                        1600 *
                        1610 *-------------------------------
                        1620 OUTL
02A7- 48                1630 TABR   PHA           CHAR IN STACK
02A8- 86 1D             1640        STX STOX      PRESERVE X-REG
02AA- 49 FF             1650        EOR #$FF      INVERT BYTE
02AC- A2 00             1660        LDX #$00      STA CHARACTER IN
02AE- 81 1B             1670        STA (PO2L,X)  INTO THE TABLE
02B0- 18                1680        CLC
02B1- A5 1B             1690        LDA PO2L      CURRENT HIGH
02B3- 69 01             1700        ADC #$1       POINTER IN
02B5- 85 1B             1710        STA PO2L
02B7- A5 1C             1720        LDA PO2H
02B9- 69 00             1730        ADC #$00
02BB- 85 1C             1740        STA PO2H
```

```
02BD- 20 E7 02 1750         JSR OOFM      ALL IN HIMEM?
02C0- 68       1760         PLA           RESTORE CHARACTER
02C1- 48       1770         PHA
02C2- 45 08    1780         EOR CSOM      WORK CHECKSUM
02C4- 85 08    1790         STA CSOM      IN
02C6- A6 1D    1800         LDX STOX      RESTORE X-REG
02C8- 68       1810         PLA           RESTORE ACC
02C9- 60       1820         RTS
               1830 *-------------------------------
               1840 *  *****  CALL 970  *****
               1850 *
               1860 * MOVE THE TABLE AT LINE IN
               1870 * BASICODE
               1880 *
               1890 *-------------------------------
02CA- A9 83    1900 UITZ    LDA #$83      END OF TEXT
02CC- 20 A7 02 1910         JSR TABR      MOVE IN TABLE
02CF- A5 08    1920         LDA CSOM
02D1- 20 A7 02 1930         JSR TABR      MOVE CSUM IN TABLE
02D4- A5 06    1940         LDA $06
02D6- 85 36    1950         STA CSWL      RESTORE
02D8- A5 07    1960         LDA $07       OUTPUT VECTOR
02DA- 85 37    1970         STA CSWH
02DC- 20 DD FB 1980         JSR PIEP      GIVE BEEP
02DF- 20 00 03 1990         JSR ZEND      OUTPUT TABLE
02E2- 20 DD FB 2000         JSR PIEP      GIVE BEEP
02E5- 20 98 03 2010         JSR VERN      VERIFY BASIC
02E8- 60       2020         RTS
               2030 *-------------------------------
               2040 *  ARE WE NEAR HIMEM?
               2050 *
               2060 *-------------------------------
02E9- C5 74    2070 OOFM    CMP $74
02EB- D0 12    2080         BNE NIET      ARE WE
02ED- A5 1B    2090         LDA PO2L      AT HIMEM?
02EF- C5 73    2100         CMP $73
02F1- D0 0C    2110         BNE NIET
02F3- A5 67    2120         LDA STBL      YES, SO OVERWRITE
02F5- 85 1B    2130         STA PO2L      BASIC PROGRAM
02F7- A5 68    2140         LDA STBH
02F9- 85 1C    2150         STA PO2H
02FB- A5 FF    2160         LDA $FF       MOVE SAME TIME FLAG
02FD- 85 FF    2170         STA FLAG
02FF- 60       2180 NIET    RTS
               2190 *-------------------------------
               2200 *
               2210 * MAKE THE BASICODE TONES
               2220 *
               2230 *-------------------------------
0300- 20 7F 03 2240 ZEND    JSR HEAD      5 SEC HEADER
0303- A5 1B    2250         LDA PO2L      MOVE END POINTER
0305- 85 1E    2260         STA PO3L      DOWN AT PO3
0307- A5 1C    2270         LDA PO2H
0309- 85 1F    2280         STA PO3H
030B- A5 19    2290         LDA PO1L
030D- 85 1B    2300         STA PO2L      MOVE CURRENT POINTER
030F- A5 1A    2310         LDA PO1H      AT BEGIN TABLE
0311- 85 1C    2320         STA PO2H
0313- A0 1D    2330         LDY #$1D
0315- A2 00    2340         LDX #$00
0317- A1 1B    2350 NBYT    LDA (PO2L,X)  READ TABLE
0319- 38       2360         SEC           STARTBIT = 1
031A- A2 0B    2370         LDX #$0B      BITCOUNTER = 11
031C- 88       2380 WAC1    DEY
031D- D0 FD    2390         BNE WAC1      TIMING LOOP 1
031F- B0 03    2400         BCS EEN1      INFO = 1?
0321- AC 20 C0 2410         LDY $C020     NO, TOGGLE OUTPUT
0324- A0 29    2420 EEN1    LDY #$29
0326- 88       2430 WAC2    DEY           TIMING LOOP 2
0327- D0 FD    2440         BNE WAC2
0329- EA       2450         NOP           TIMING CORRECTION
032A- AC 20 C0 2460         LDY $C020     TOGGLE OUTPUT
032D- A0 29    2470         LDY #$29
032F- 88       2480 WAC3    DEY
0330- D0 FD    2490         BNE WAC3      TIMING LOOP 3
0332- B0 03    2500         BCS EEN2      INFO = 1?
0334- AC 20 C0 2510         LDY $C020     NO, TOGGLE OUTPUT
0337- A0 29    2520 EEN2    LDY #$29
0339- 88       2530 WAC4    DEY
033A- D0 FD    2540         BNE WAC4      TIMING LOOP 4
033C- EA       2550         NOP           TIMING CORRECTION
033D- AC 20 C0 2560         LDY $C020     TOGGLE OUTPUT
0340- A0 27    2570         LDY #$27      PRESET TIMING LOOP
0342- 4A       2580         LSR           SHIFT BIT IN CARRY
0343- CA       2590         DEX           DECREMENT BITCOUNTER
0344- F0 03    2600         BEQ NEXT      NEXT BYTE
0346- EA       2610         NOP           TIMING CORRECTION
0347- D0 D3    2620         BNE WAC1      JUMP BACK
0349- A0 1E    2630 NEXT    LDY #$1E
034B- 18       2640         CLC
034C- A5 1B    2650         LDA PO2L      CURRENT HIGH
034E- 69 01    2660         ADC #$1       POINTER IN
0350- 85 1B    2670         STA PO2L
0352- A5 1C    2680         LDA PO2H
0354- 69 00    2690         ADC #$00
0356- 85 1C    2700         STA PO2H
0358- C5 74    2710         CMP $74       ALL FIT
035A- EA       2720         NOP           IN HIMEM
035B- D0 13    2730         BNE CHCK      NO
035D- A0 1C    2740         LDY #$1C
035F- A5 1B    2750         LDA PO2L
0361- C5 73    2760         CMP $73
0363- D0 0B    2770         BNE CHCK      NO
0365- EA       2780         NOP           WE ARE AT
0366- A0 19    2790         LDY #$19      HIMEM
0368- A5 67    2800         LDA STBL      MOVE CURRENT
036A- 85 1B    2810         STA PO2L      POINTER TO
036C- A5 68    2820         LDA STBH      BEGIN MEMORY
036E- 85 1C    2830         STA PO2H
0370- A5 1C    2840 CHCK    LDA PO2H
0372- C5 1F    2850         CMP PO3H      ARE WE ARE THE
0374- D0 A1    2860         BNE NBYT      END OF THE TABLE?
0376- 88       2870         DEY           CORRECT TIME
0377- 88       2880         DEY
0378- 88       2890         DEY
0379- A5 1B    2900         LDA PO2L
037B- C5 1E    2910         CMP PO3L      COMPARE CURRENTS
037D- D0 98    2920         BNE NBYT
               2930 *-------------------------------
               2940 *
               2950 * HEADER AND TRAILER ROUTINE
               2960 * TONE OF 2400 HZ DURING 5 SEC
               2970 *
               2980 *-------------------------------
037F- A0 1C    2990 HEAD    LDY #$1C      GENERATE TONE
0381- A2 5D    3000         LDX #$5D      OF 2400 HZ
0383- 86 1D    3010         STX STOX      DURING 5 SEC
0385- 88       3020 WAC5    DEY
0386- D0 FD    3030         BNE WAC5      TIMING LOOP5
0388- AC 20 C0 3040         LDY $C020     TOGGLE OUTPUT
038B- EA       3050         NOP
038C- A0 28    3060         LDY #$28
038E- CA       3070         DEX
038F- D0 F4    3080         BNE WAC5
0391- A0 26    3090         LDY #$26
0393- C6 1D    3100         DEC STOX
0395- D0 EE    3110         BNE WAC5      TONE STILL NOT FINSIHED
0397- 60       3120         RTS           END TONE
               3130 *-------------------------------
               3140 *
               3150 * IS BASIC OVERWRITTEN THROUGH TABLE?
               3160 *
               3170 *-------------------------------
0398- A5 FF    3180 VERN    LDA FLAG      IS BASIC OVERWRITTEN
039A- C9 FF    3190         CMP #$FF
039C- D0 15    3200         BNE EIND      NO
039E- A5 67    3210         LDA STBL      YES, THEN DESTROY
03A0- 85 AF    3220         STA $AF       ALSO GOOD
```

```
03A2- 85 69    3230        STA $69
03A4- A5 68    3240        LDA STBH
03A6- 85 B0    3250        STA $B0
03A8- 85 6A    3260        STA $6A
03AA- A0 00    3270        LDY #$00
03AC- A9 00    3280        LDA #$00
03AE- 91 67    3290        STA (STBL),Y
03B0- C8       3300        INY
03B1- 91 67    3310        STA (STBL),Y
03B3- 60       3320 EIND   RTS
               3330        .OR $3C0
03C0- 4C 80 02 3340        JMP INIT     CALL 960-MOVE OUTPUT
               3350        .OR $3C5     CALL 965-NONSENSE
03C5- 4C 2D FF 3360        JMP PERR
               3370        .OR $3CA
03CA- 4C CA 02 3380        JMP UITZ     CALL 970-MOVE IN LINE
:ASM

               1010 *--------------------------------
               1020 * BASICODE READ ROUTINE
               1030 *    FOR THE APPLE II
               1040 *
               1050 *
               1060 * VERSION  SEPT 27, 1981
               1070 *
               1080 *--------------------------------
               1090 *
               1100 * by J.Hermann Peojhm
               1110 *
               1120 *--------------------------------
               1130        .OR $283
0036-          1140 CSWL   .EQ $36
0037-          1150 CSWH   .EQ $37
0038-          1160 KSWL   .EQ $38
0039-          1170 KSWH   .EQ $39
00AF-          1180 EOBL   .EQ $AF
00B0-          1190 EOBH   .EQ $B0
0019-          1200 PO1L   .EQ $19
001A-          1210 PO1H   .EQ $1A
001B-          1220 PO2L   .EQ $1B
001C-          1230 PO2H   .EQ $1C
001E-          1240 PO3L   .EQ $1E
001F-          1250 PO3H   .EQ $1F
002E-          1260 CSOM   .EQ $2E
001D-          1270 COUN   .EQ $1D
002F-          1280 LAST   .EQ $2F
FDF6-          1290 COUZ   .EQ $FDF6
               1300 *
0283- A5 AF    1310 INIT   LDA EOBL     MOVE POINTER PO2
0285- 85 1B    1320        STA PO2L     AT THE START OF TABLE
0287- A5 B0    1330        LDA EOBH
0289- 18       1340        CLC
028A- 69 01    1350        ADC #$01
028C- C5 74    1360        CMP $74      ABOVE HIMEM?
028E- B0 3F    1370        BCS STOP     YES, THEN STOP
0290- 85 1C    1380        STA PO2H
0292- A9 00    1390        LDA #$00
0294- 85 2E    1400        STA CSOM     CHECKSUM=0
0296- 85 1D    1410        STA COUN
0298- 20 01 03 1420 FIND   JSR RTYD     LOOK FOR START TONE
029B- C6 1D    1430        DEC COUN     COUNT NUMBER OF PERIOI
029D- D0 F9    1440        BNE FIND     ALL 255?
029F- A0 1B    1450 BEGIN  LDY #$1B
02A1- 20 04 03 1460 STAR   JSR RBIT     LOOK FOR STARTBIT
02A4- 90 F9    1470        BCC BEGIN    FOUND?
02A6- 20 04 03 1480        JSR RBIT     WATCH HALF PERIOD
02A9- A0 30    1490        LDY #$30     0/1 CRITERION
02AB- 20 ED 02 1500        JSR RBYT     READ BYTE
02AE- 81 1B    1510        STA (PO2L,X) MOVE BYTE IN TABLE
02B0- 49 FF    1520        EOR #$FF     INVERT BYTE
02B2- 8D 27 04 1530        STA $0427    MOVE TO SCREEN
02B5- 45 2E    1540        EOR CSOM     DETERMINE NEW
02B7- 85 2E    1550        STA CSOM     CHECKSUM
02B9- F0 15    1560        BEQ CEND     =0? THEN CHECK IF WE ARE AT END
02BB- A0 17    1570        LDY #$17     0/1 CRITERION
02BD- E6 1B    1580 ADD1   INC PO2L     INCREMENT CURRENT
02BF- D0 02    1590        BNE HIM
02C1- E6 1C    1600        INC PO2H
02C3- A5 1C    1610 HIM    LDA PO2H     ARE WE IN HIMEM?
02C5- C5 74    1620        CMP $74
02C7- D0 D8    1630        BNE STAR
02C9- A5 1B    1640        LDA PO2L
02CB- C5 73    1650        CMP $73
02CD- D0 D2    1660        BNE STAR
02CF- 60       1670 STOP   RTS          STOP, WE ARE IN HIMEM
02D0- 38       1680 CEND   SEC          CHECK FOR EOT SIGNAL
02D1- A5 1B    1690        LDA PO2L
02D3- E9 01    1700        SBC #$01     DECREMENT THE TABLE POINTER
02D5- 85 1B    1710        STA PO2L     BY 1
02D7- A5 1C    1720        LDA PO2H
02D9- E9 00    1730        SBC #$00
02DB- 85 1C    1740        STA PO2H
02DD- A0 14    1750        LDY #$14     0/1 CRITERION
02DF- A1 1B    1760        LDA (PO2L,X) READ CHARACTER
02E1- C9 7C    1770        CMP #$7C     =EOT? (INVERSE)
02E3- F0 2E    1780        BEQ TABL     YES, FINISHED, READ IN AS BASIC
02E5- E6 1B    1790        INC PO2L     NO
02E7- D0 D4    1800        BNE ADD1     INCREMENT THE POINTER
02E9- E6 1C    1810        INC PO2H     AGAIN BY 1
02EB- D0 D0    1820        BNE ADD1     AND JUMP BACK
               1830 *--------------------------------
               1840 *
               1850 * READ BYTE
               1860 *
               1870 *--------------------------------
02ED- A2 08    1880 RBYT   LDX #$08     BIT COUNTER=8
02EF- 48       1890 RBY2   PHA          SAVE ACC
02F0- 20 01 03 1900        JSR RTYD     READ BIT (1 IF 0)
02F3- B0 04    1910        BCS ROTA     WAS A NULL:JUMP
02F5- 20 01 03 1920        JSR RTYD     START YET A PERIOD
02F8- 18       1930        CLC          BUT DO NOT DECODE
02F9- 68       1940 ROTA   PLA          GET ACC AGAIN
02FA- 6A       1950        ROR          WRITE CARRY WITHIN
02FB- A0 2E    1960        LDY #$2E     0/1 CRITERION
02FD- CA       1970        DEX          DECREMENT BIT COUNTER
02FE- D0 EF    1980        BNE RBY2     FINISHED? NO, JUMP
0300- 60       1990        RTS          BYTE COMPLETE IN ACC
0301- 20 04 03 2000 RTYD   JSR RBIT     START TIME V.E. PERIOD
0304- 88       2010 RBIT   DEY          DECREMENT TIME COUNTER
0305- AD 60 C0 2020        LDA $C060    READ CASSETTE INPUT
0308- 45 2F    2030        EOR LAST     IS IT CHANGED?
030A- 10 F8    2040        BPL RBIT     NO, BACK
030C- 45 2F    2050        EOR LAST     TURN AROUND AGAIN
030E- 85 2F    2060        STA LAST     AND SAVE LAST INPUT IN
0310- C0 80    2070        CPY #$80     TIME COUNTER SET/RESET CARRY
0312- 60       2080        RTS
0313- A5 AF    2090 TABL   LDA EOBL
0315- 85 19    2100        STA PO1L     POINTER PO1 AT
0317- A5 B0    2110        LDA EOBH     START MOVE TABLE
0319- 85 1A    2120        STA PO1H
031B- E6 1A    2130        INC PO1H
031D- A5 38    2140        LDA KSWL     IN & OUTPUT
031F- 85 08    2150        STA $08      VECTOR STORAGE
0321- A5 39    2160        LDA KSWH
0323- 85 09    2170        STA $09
0325- A5 36    2180        LDA CSWL
0327- 85 06    2190        STA $06
0329- A5 37    2200        LDA CSWH
032B- 85 07    2210        STA $07
032D- A9 3E    2220        LDA #NKAR    BASIC HALT INPUT
032F- 85 38    2230        STA KSWL     FROM ROUTINE NKAR
0331- A9 03    2240        LDA /NKAR
0333- 85 39    2250        STA KSWH
0335- A9 F0    2260        LDA #$F0     OUTPUT TO SCREEN
0337- 85 36    2270        STA CSWL
0339- A9 FD    2280        LDA #$FD
033B- 85 37    2290        STA CSWH
```


CONT ON PAGE 16



Who is able to hop on a trampoline, catch a swinging trapeze, keep his balance on a flying carpet, and run fearlessly through falling plungers? Who looks like the 'San Diego chicken' and has a victory dance to match? Who gets his wig whirled whenever he makes a false move? No one else, but SAMMY LIGHTFOOT, the star of a new game which bears his name.

There are three different scenarios in SAMMY LIGHTFOOT. Each scenario has twelve different levels of difficulty. The object of the game is to reach top of each scenario by making Sammy perform his tricks. You start at the bottom with a bonus score of 9999 and as you take each step the bonus score ticks away. If you sucessfully reach the top of all three scenarios, you start back at the bottom on scenario 1, but this time at an increased level of difficulty.

In the first scenario you must use a trampoline to jump high enough to reach a platform, jump off the platform onto another trampoline. Now grab hold of the trapeze and swing across the chasm. But watch out for the bouncing balls. If they hit you, you'll have to start over at the bottom. When and if you finally reach the top platform, you have to avoid colliding with a crazy pumpkin who wears sunglasses.

The second scenario requires that you make your way across a series of platforms while dodging some rising and falling plungers. Be careful not to stay in one spot too long because the platforms have a nasty habit of disappearing. Once you reach the other side you take an elevator ride up and then you travel back on top of the plungers. Another elevator ride, and you are whisked off on a flying carpet. If you're not careful the carpet will go left while you're still going right and it's goodby Sammy.

When you're finally able to face the challenges of the third scenario, you'll find some very annoying puff balls and several falling hammers guard the path to the grand finale; a trapeze swing across a huge flaming pit.

SAMMY LIGHTFOOT can be played with the keyboard, paddles, or a joystick. I found the joystick the easiest to use. The keyboard version is designed so that left and right-handed players can play equally well. The top ten scores are saved on disk and you have the option to reset the high scores whenever you want. You can also select to start at a higher level of difficulty so that experienced players may find play continously challenging.

My first reaction to SAMMY LIGHTFOOT was that with only three levels of play, it would not keep my attention for long. What I didn't realize at the time was the fact that the different levels of difficulty require you not only to play better but you must also use varying strategies to be successful. I spent more time playing it than most games and I still haven't gotten past the fourth level of difficulty. What I liked best about the game was that although you are racing against the clock for a high score, the best playing strategy proved to be one of patience, waiting for just the right moment to make a move.

SAMMY LIGHTFOOT was written by Warren Schwader and is distributed by Sierra On-Line.

AN INTRODUCTION TO ARRAYS
by John Wrenholt

Do you want to know a secret method that will allow you to write programs faster. This secret will also make your programs shorter and easier to 'read'. The method is not really secret, but somehow many beginning programmers don't learn about it. As you can probably guess from the title of this article, the method involves using arrays in your programs.

What is an array? Well, an array is simply a method of defining and using sets of related variables. Just as there are three types of simple variables, there are three types of array variables; reals, integers, and strings.

Suppose for example, that we wanted to write a program which would simulate the rolling of one die. The object of the program would be to tally the number of occurances of each face on the die in one thousand rolls.

Here's how that program would look without the use of arrays:

```
10  TEXT : HOME : HTAB 5 : VTAB 4
20  PRINT "DIE ROLLING SIMULATION"
30  S1=0: S2=0: S3=0: S4=0: S5=0: S6=0
40  FOR X = 1 TO 1000
50  ROLL = INT (RND(1)*6) + 1
60  HTAB 20 : VTAB 10 : PRINT ROLL
70  IF ROLL = 1 THEN S1 = S1 + 1
80  IF ROLL = 2 THEN S2 = S2 + 1
90  IF ROLL = 3 THEN S3 = S3 + 1
100 IF ROLL = 4 THEN S4 = S4 + 1
110 IF ROLL = 5 THEN S5 = S5 + 1
120 IF ROLL = 6 THEN S6 = S6 + 1
130 NEXT X
150 HTAB 5 : PRINT "1 OCCURRED ";S1;"
TIMES"
160 HTAB 5 : PRINT "2 OCCURRED ";S2;"
TIMES"
170 HTAB 5 : PRINT "3 OCCURRED ";S3;"
TIMES"
180 HTAB 5 : PRINT "4 OCCURRED ";S4;"
TIMES"
190 HTAB 5 : PRINT "5 OCCURRED ";S5;"
TIMES"
200 HTAB 5 : PRINT "6 OCCURRED ";S6;"
TIMES"
```

Now let's rewrite the same program using arrays. It would look this this:

```
10  TEXT : HOME : VTAB 5 : HTAB 5
20  PRINT "DIE ROLLING SIMULATION"
30  DIM S(6)
40  FOR X = 1 TO 1000
50  ROLL = INT (RND(1)*6) + 1
60  VTAB 10 : HTAB 20 : PRINT ROLL
70  S(ROLL) = S(ROLL) + 1
130 NEXT X
140 VTAB 10
150 FOR X = 1 TO 6
160 HTAB 5
170 PRINT X;" OCCURRED ";S(X);" TIMES"
180 NEXT X
```

Other than being somewhat shorter, what else has been changed in the second program? The first change is in line 30 where we defined the variables. To define an array we must tell Applesoft how many elements to allocate for the entire array. This is done with a DIM statement. We say that we have DIMensioned an array. We have created seven new variables called S(0), S(1), S(2), S(3), S(4), S(5), and S(6). Notice that arrays always start with element zero (0). As the programmer, you can choose to use the zero element or ignore it as we have done in the example program.

The next change in the program occurs in lines 70-120. We were able to replace all these lines with just one line (70 S(ROLL) = S(ROLL) + 1). We also were able to shorten the code in lines 150-200.

In order to select an element from the array, we have to give it a distinct name. We do this by using a subscript: that is, to select the sixth element of the array, we would use S(6). S is the array name and 6 which we enclose in parenthesis is the subscript. Subscripts can be either constants or variables. In our program instead of using the constant value of 6 for the subscript, we are using the variable ROLL to select the desired element. When using variables for subscripts you must be careful to ensure that the value of the variable is a valid subscript for the array. If you

use a value which is larger than the size of your array, a BAD SUBSCRIPT error will be given.

Another warning to keep in mind when working with arrays is this: If you use an array variable such as A(6) in your program before the array has been DIMensioned, Applesoft will automatically create an array with eleven elements (subscripts 0 thru 10). Then when you try to DIMension your array, Applesoft will give you a REDIMENSIONED ERROR. For this reason, DIM statements are usually placed near the beginning of the program.

If you type in both versions and RUN them you will notice another difference. It takes approximately 45 seconds for the first version to run, while the second version runs in only 33 seconds. This is not to say that Applesoft actually processes arrays any faster than simple variables. The time savings are due to the absence of the IF statements in lines 70 through 120.

In our next month's article on arrays we will discuss multi-dimensional arrays.

TIP (.) - THE INVOICE PRINTER

by James T. DeMay Jr.

Reprinted from WASHINGTON APPLE PI, JULY 1983

When operating a business, the need arises for a system for storing, retrieving and printing order information. TIP(.) will handle an order from placement to final collection and then some.

Customer name, street address, city, state, zip code, and telephone number are maintained in the CUSTOMER FILE. It is a random access file, having the lengths; 30, 30, 30, 2, 5, and 12 respectively. This plus one additional space for a carriage return for each field, equals a total record length of 115. The specific data for each order is maintained in the INVOICE FILE. This file is built from all the information entered for each order. The number of invoices and the particular customer to whom each invoice was issued is kept in record #0. The length of each record in this random access file is 575, a rather large file, but necessary if all data from each invoice is to be retained. The AR-N-FILEs contain the billing information. These Accounts Receivable files are created when customer information is entered. They are automatically updated whenever an invoice is prepared for each customer. TIP(.) can figure finance charges based on the overdue amount, or on a specified amount, add the finance charges to the outstanding amount, and then print a statement which can be sent as a payment notice. When a payment is received, TIP(.) will record the payment, subtract the payment from the amount outstanding, and save the information to the accounts receivable file. A separate AR-N-FILE is maintained for each customer. To save disk space, this is a sequential text file. The N is a number which is the actual customer number assigned on entry of customer name, and pertinent data.

A unique invoice number is composed of two four digit numbers joined by a dash. The first four digits equal the customer number as in the AR-N-FILE. The second part of the invoice number is the actual record number in the INVOICE FILE. Using this system, one can always be aware of the customer number and the invoice record number.

Customer information need only be entered once, although it can be modified at any time. Thereafter, all that is required to access the data for a particular customer is a number. If you forget which number is assigned to which customer, TIP(.) will be glad to help you remember. Just press the V for View customer file, and each customer name and number will be presented at the bottom of the screen. To stop this loop and return to entering data, just press the SPACE bar.

Since the files used by TIP(.) are so large, I suggest using a special data disk which has DOS removed thus allowing more room for data storage. There are several public domain programs available which will do this.

USING THE PROGRAM

TIP(.) is menu driven, and prompts when input is required. When first run, TIP(.) presents the title page and requests that you insert the data disk. Pressing any key causes the program to attempt to read the CUSTOMER FILE and the INVOICE FILE. If no files exist, TIP(.) will ask you to enter information for customer #1. After the data has been provided, the "Input invoice data" section is entered. Here you have a choice of returning to the menu, adding a new customer, changing existing customer information, or entering the appropriate number for a customer to place an order. Entering a legitimate number directs TIP(.) to read the customer information from the disk and display it on the screen. You are then asked where to ship the merchandise. If the "ship to" address is the same as the "sold to" address, a RETURN is all that is required to complete the "ship to" section of the invoice. If the addresses are different, the appropriate information must be entered.

The ship date, terms, and carrier are requested in order, each having their default values in parentheses. Pressing RETURN will cause the default values to be entered in the correct positions. This completes the top portion of the invoice form. Pressing any key will present the remainder of the form and the appropriate input prompts.

Up to ten items, quantities and prices can be entered. TIP(.) asks for the item, quantity, price per unit, and requests that you verify the data just entered. Numerical values are expected for quantity and price. Pressing any key except N will be interpreted as an affirmative answer. The subtotal for each item is computed and the data displayed. TIP(.) continues to step through the loop until ten items have been entered. What if there are less than ten items? Just pressing RETURN for the item will end the loop and move to the final section of the invoice. The subtotals of all items previously entered are added and displayed. Additional information required to complete the invoice is requested; such as discount, shipping and handling fee, and sales tax. The total bill is computed and you are given the option of saving the data, or aborting this entry. If abort is selected, TIP(.) erases the data entered, and returns to the main menu. If the save option is chosen, an entry is made in the INVOICE FILE, and the AR-N-FILE. You are then asked if the invoice is to be printed at this time. This data can be recalled at any time from the menu by selecting the Print invoice option. The invoice will be printed to the screen, and then if desired, to a printer. As written, this option requires a printer in slot #1. More about printing later.

Reconciling an accounts receivable file is done by choosing the Update/verify account option. From here, you choose a customer by number, then compute finance charges if required. Enter a payment and then if desired, print a statement to be sent to the customer as a payment notice.

Choosing the Totals to date option requests the printing of each customer number, name, and important information from each invoice sent to that customer. Totals are printed following the last invoice for each customer. The program steps through the customer file and then through the invoice file, selecting the appropriate records, and then printing them to the printer only. This continues until all invoices for each customer have been printed. A final tally of the totals of all columns is printed for all customers.

DECIMAL FORMATTING AND PRINT ROUTINES

I first saw this decimal formatting technique in the March 1980 issue of SOFTSIDE. It has been adapted to work with TIP(.). The number to format is passed to the subroutine at line #500 as the variable N. The number to print is returned in N$ with the length in

contd.

Listing of TIP(.) - James T. DeMay Jr.

```
 1 ONERR  GOTO 30000
 3 GOTO 10000
 4 VTAB CV + 2: CALL  - 958: VTAB 22: RETURN
 5 VTAB 23: CALL  - 958: RETURN
 6 GOSUB 5: HTAB 5: PRINT "PRESS ANY KEY TO CONTINUE
   ";: GET CH$: PRINT CH$: RETURN
10 PW = 0:FL = 0:OD = 0:TL = 0:AM = 0:SL = 0:SF = 0:ST
   = 0:NI = 0:SD$ = "":TS$ = "":CR$ = "": FOR K = 1 TO
   10:I$(K) = "": NEXT K
15 TX = 0
20 FOR C = 1 TO 7:LL(C) = 0:TT(C) = 0: NEXT C: GOTO
   5000
30 FOR K = 1 TO 13
33 VTAB V: HTAB 9 + K
36 GET Q$
37 IF Q$ =  CHR$ (8) THEN K = K - 1: IF K < 1 THEN K =
   1
38 IF K = 13 THEN Q$ = "": FOR X = 1 TO 12:Q$ = Q$ +
   QQ$(X): NEXT : GOTO 54
39 IF K = 1 AND Q$ =  CHR$ (13) THEN Q$ = " ":K = 13:
   GOTO 38
42 IF Q$ =  CHR$ (8) THEN  PRINT Q$;" ": GOTO 33
45 IF K = 3 THEN Q$ = Q$ + "/":K = K + 1
48 IF K = 7 THEN Q$ = Q$ + "-":K = K + 1
50 IF  ASC (Q$) < 48 OR  ASC (Q$) > 57 THEN Q$ =  CHR$
   (8): GOTO 42
51 PRINT Q$:QQ$(K) = Q$
54 NEXT K
55 C$(6) = Q$:Q$ = ""
60 RETURN
80 N
90 : REM THIS LINE IS REFERENCED TO ENTER THE CLOCK
   ROUTINE. KEEP THIS LINE AS <90:> TO  SAVE SPACE.
91 PRINT D$;"PR#3"
92 PRINT D$;"IN#3"
93 INPUT "%";W$
94 PRINT D$;"PR#0"
95 PRINT D$;"IN#0"
96 DA$ =  LEFT$ (W$,10)
97 RETURN
99 :
100  REM USE LINES 91 THRU 97 IF
101  REM YOU HAVE A CLOCK CARD
102  REM IN SLOT #3.  LINES 105
103  REM THRU 108 IF YOU DON'T
104  :
105  HOME : VTAB 23: PRINT "ENTER TODAY'S DATE AS (MON
     JUN 26):": VTAB 24: HTAB 12: PRINT "_____ ___";:
     VTAB 24: HTAB 12: INPUT "";DA$
106  IF  LEN (DA$) < 10 OR  LEN (DA$) > 10 THEN  PRINT
     G$;: GOTO 105
107  W$ = DA$
108  RETURN
109  :
190  IF  LEN (CH$) > 1 THEN C$(CH) = CH$:CH = 0: GOTO
     880
192  HTAB 10: PRINT C$(CH)
198  RETURN
500  IF N >  = .01 THEN 510
501  IF N <  = 0 THEN 510
503  N = N * 1000
504  N$ = ".00" +  STR$ (N)
506  NL = 2
508  RETURN
510  N =  INT (N * 100 + .5) / 100
512  N$ =  STR$ ( INT (N)):NL =  LEN (N$) * ( VAL (N$)
     <  > 0):N$ =  STR$ (N)
514  IF  INT (N) = N THEN N$ = N$ + Z2$
516  IF 10 * N =  INT (10 * N) THEN N$ = N$ + Z1$
518  IF  LEN (N$) < 3 THEN N$ = N$ + Z1$
520  Z5$ =  RIGHT$ (N$,2)
522  Z6$ =  LEFT$ (Z5$,1)
524  IF Z6$ = "." THEN N$ = N$ + Z1$
526  NL =  LEN (N$) - 1
528  RETURN
599  REM ** PRINT FORMATTER **
600  T = 0: IF N < 10 THEN T = 1: GOTO 620
605  IF N < 100 THEN T = 0: GOTO 620
610  IF N < 1000 THEN T =  - 1: GOTO 620
615  IF N < 10000 THEN T =  - 2: GOTO 620
620  POKE 36,H + T: PRINT N;: RETURN
650  IF N < 10 THEN  PRINT "000";: RETURN
655  IF N < 100 THEN  PRINT "00";: RETURN
660  IF N < 1000 THEN  PRINT "0";: RETURN
700  SH$(1) = CH$: IF  LEN (CH$) > 30 THEN 6160
705  VTAB 11: HTAB 10: PRINT SH$(1)
710  GOSUB 5: INPUT "STREET: ";SH$(2): IF  LEN
     (SH$(2)) > 30 THEN 710
715  VTAB 13: HTAB 10: PRINT SH$(2)
720  GOSUB 5: INPUT "CITY: ";SH$(3): IF  LEN (SH$(3))
     > 30 THEN 720
725  VTAB 15: HTAB 10: PRINT SH$(3)
730  GOSUB 5: INPUT "STATE: ";SH$(4): IF  LEN (SH$(4))
     > 2 THEN 730
735  VTAB 15: HTAB 38: PRINT SH$(4)
740  GOSUB 5: INPUT "ZIP CODE: ";SH$(5): IF  LEN
     (SH$(5)) > 5 THEN 740
745  VTAB 17: HTAB 10: PRINT SH$(5)
750  GOSUB 5: PRINT "PHONE # ";:V = 23: GOSUB
     30:SH$(6) = C$(6)
755  VTAB 17: HTAB 28: PRINT SH$(6)
760  GOTO 6210
800  TEXT : HOME : IF CN% > TQ% + 99 THEN  VTAB 10:
     INVERSE : PRINT "TO ENTER MORE CUSTOMERS YOU MUST
     EXIT  THE PROGRAM AND THEN RERUN IT. A MAXIMUM
     100 CUSTOMERS CAN BE ADDED AT ONE TIME": NORMAL :
     FOR K = 1 TO 2000: NEXT K: GOTO 10
801  IF CH$ = "A" THEN 808
802  TEXT : HOME : PRINT "<M>ENU, OR ENTER CUSTOMER #1
     TO ";CN%;" ";: INPUT "";CH$
803  IF CH$ = "M" THEN 10
804  CH =  VAL (CH$): IF CH < 1 OR CH > CN% THEN 800
805  C% = CH: GOSUB 1100:HE = 1
806  TEXT :: HOME : IF HE THEN  GOSUB 1100: GOTO 810
808  TEXT : HOME :CN% = CN% + 1: PRINT "THIS WILL BE
     CUSTOMER #";CN%:C% = CN%
810  VTAB 5: HTAB 5: PRINT "NAME: ";: HTAB 10: CALL  -
     958: PRINT US$;: IF CN% > 1 THEN  VTAB 7: HTAB 5:
     PRINT "PRESS RETURN FOR MENU ...": IF HE THEN
     VTAB 5: HTAB 10: PRINT C$(1): GOTO 820
811  IF HE THEN  VTAB 5: HTAB 10: PRINT C$(1): GOTO
     820
812  VTAB 5: HTAB 10: INPUT "";C$(1): IF CN% > 1 THEN
     IF C$(1) = "" THEN CN% = CN% - 1: POP : GOTO 10
813  IF CN% = 1 AND C$(1) = "" THEN 810
815  IF  LEN (C$(1)) > 30 THEN  PRINT G$;: GOTO 810
820  VTAB 7: HTAB 3: PRINT "STREET: ";: HTAB 10: CALL
     - 958: PRINT US$;: IF HE THEN  HTAB 10: PRINT
     C$(2): GOTO 830
822  VTAB 7: HTAB 10: INPUT "";C$(2)
825  IF  LEN (C$(2)) > 30 THEN  PRINT G$;: GOTO 820
830  VTAB 9: HTAB 5: PRINT "CITY: ";: HTAB 10: CALL  -
     958: PRINT US$;: IF HE THEN  HTAB 10: PRINT
     C$(3): GOTO 840
832  VTAB 9: HTAB 10: INPUT "";C$(3)
835  IF  LEN (C$(3)) > 27 THEN  PRINT G$;: GOTO 830
840  VTAB 11: HTAB 4: PRINT "STATE: ";: HTAB 10: CALL
     - 958: PRINT  CHR$ (95); CHR$ (95);: IF HE THEN
     HTAB 10: PRINT C$(4): GOTO 850
842  VTAB 11: HTAB 10: INPUT "";C$(4)
845  IF  LEN (C$(4)) > 2 THEN  PRINT G$;: GOTO 840
850  VTAB 13: HTAB 1: PRINT "ZIP CODE: ";: HTAB 10:
     CALL  - 958: PRINT UZ$;: IF HE THEN  HTAB 10:
     PRINT C$(5): GOTO 870
852  VTAB 13: HTAB 10: INPUT "";C$(5)
855  IF C$(5) = "" THEN 870
860  IF  LEN (C$(5)) <  > 5 THEN  PRINT G$;: GOTO 850
865  IF  VAL (C$(5)) < 1 THEN  PRINT G$: GOTO 850
870  VTAB 15: HTAB 2: PRINT "PHONE #:";: HTAB 10: CALL
     - 958: PRINT UP$;: IF HE THEN  HTAB 10: PRINT
     C$(6): GOTO 880
872  VTAB 15: HTAB 10:V = 15: GOSUB 30
880  PRINT : VTAB 20: PRINT "<A> BORT, <C> ORRECT, OR
     <S> AVE ";: GET CH$: PRINT CH$:CH = 0
882  IF HE THEN  IF CH$ = "A" THEN HE = 0: GOTO 6005
885  IF CH$ = "A" THEN CN% = CN% - 1: GOTO 6005
```

contd.

```
887  IF CH$ = "C" THEN 905
890  IF CH$ = "S" THEN  POKE 34,21: VTAB 20: CALL  -
     958: HTAB 14: INVERSE : PRINT "SAVING DATA":
     NORMAL : GOSUB 1200:CF$(C%) = C$(1):CH = C%: IF
     HE THEN HE = 0: GOTO 6005
895  IF CH$ = "S" THEN TZ = 0: GOSUB 2000: GOTO 6005
900  GOTO 880
905  VTAB 20: CALL  - 958: PRINT "PRESS <RETURN> TO
     POSITION CURSER ";
910  INPUT "";CH$:CH = 1: VTAB 5: HTAB 10: INPUT
     "";CH$: VTAB 5: GOSUB 190
915  IF  LEN (CH$) < 1 THEN CH = CH + 1: VTAB 7: HTAB
     10: INPUT "";CH$: VTAB 7: GOSUB 190
920  IF  LEN (CH$) < 1 THEN CH = CH + 1: VTAB 9: HTAB
     10: INPUT "";CH$: VTAB 9: GOSUB 190
925  IF  LEN (CH$) < 1 THEN CH = CH + 1: VTAB 11: HTAB
     10: INPUT "";CH$: VTAB 11: GOSUB 190
930  IF  LEN (CH$) < 1 THEN CH = CH + 1: VTAB 13: HTAB
     10: INPUT "";CH$: VTAB 13: GOSUB 190
935  IF  LEN (CH$) < 1 THEN CH = CH + 1:V = 15: GOSUB
     30
940  GOTO 880
1000  PRINT D$;"OPEN CUSTOMER FILE,L115"
1010  PRINT D$;"READ CUSTOMER FILE,R0"
1020  INPUT CN%
1040  PRINT D$;"CLOSE"
1050  RETURN
1100  PRINT D$;"OPEN CUSTOMER FILE,L115"
1110  PRINT D$;"READ CUSTOMER FILE,R";C%
1120  FOR I = 1 TO 6
1122  INPUT C$(I)
1124  NEXT I
1130  PRINT D$;"CLOSE"
1140  RETURN
1200  PRINT D$;"OPEN CUSTOMER FILE,L115"
1210  PRINT D$;"WRITE CUSTOMER FILE,R0"
1212  PRINT CN%
1215  PRINT D$;"CLOSE"
1216  IF FL THEN FL = 0: RETURN
1217  PRINT D$;"OPEN CUSTOMER FILE,L115"
1218  PRINT D$;"WRITE CUSTOMER FILE,R"C%
1220  FOR I = 1 TO 6: PRINT C$(I): NEXT I
1230  PRINT D$;"CLOSE"
1240  RETURN
1300  CF = 1
1330  PRINT D$;"OPEN CUSTOMER FILE,L115"
1340  FOR K = 1 TO CN%
1350  PRINT D$;"READ CUSTOMER FILE,R";K
1360  INPUT CF$(K): NEXT K
1370  PRINT D$;"CLOSE"
1380  RETURN
1400  PRINT D$;"OPEN INVOICE FILE-";YR$;",L575"
1410  PRINT D$;"WRITE INVOICE FILE-";YR$;",R";0
1415  PRINT IN%
1420  FOR K = 1 TO IN%
1430  PRINT B(K)
1440  NEXT K
1450  PRINT D$;"CLOSE"
1460  RETURN
1500  PRINT D$;"OPEN INVOICE FILE-";YR$;",L575"
1510  PRINT D$;"READ INVOICE FILE-";YR$;",R";0
1515  INPUT IN%
1517  IF  NOT D THEN  DIM B(IN% + 100):D = 1
1520  FOR K = 1 TO IN%
1530  INPUT B(K)
1540  NEXT K
1550  PRINT D$;"CLOSE"
1560  RETURN
1900  GOSUB 2100
1910  TZ = TZ + 1
1920  DE$(TZ) =  LEFT$ (SD$,6)
1930  CC$(TZ) = "D": REM DEBIT
1940  CCE%(TZ) = I%: REM INVOICE #
1950  AM(TZ) = TL: REM TOTAL DUE
1980  BD(TZ) = BD(TZ - 1) + TL
1985  BE = BD(TZ)
1990  GOSUB 2000
2000  PRINT D$;"OPEN AR-";CH;"-FILE"
2010  PRINT D$;"WRITE AR-";CH;"-FILE"
2020  PRINT BE: REM BALANCE
2030  PRINT TZ: REM   # TRANSACTIONS
2033  IF TZ = 0 THEN TZ = 1
2035  FOR X = 1 TO TZ
2040  PRINT DE$(X): REM DATE OF TRANSACTION
2050  PRINT CC$(X): REM  CREDIT FLAG =P IF PAYMENT
2060  PRINT CE%(X): REM  INVOICE OR CK#
2070  PRINT AM(X): REM AMOUNT OF TRANSACTION
2080  PRINT OD(X):: REM AMOUNT OVERDUE
2090  PRINT FC(X): REM FINANCE CHARGE
2092  PRINT BD(X): REM  BALANCE DUE
2093  NEXT X
2094  PRINT DA$
2095  PRINT D$;"CLOSE"
2099  RETURN
2100  PRINT D$;"OPEN AR-";CH;"-FILE"
2110  PRINT D$;"READ AR-";CH;"-FILE"
2120  INPUT BE: REM  BALANCE
2130  INPUT TZ: REM  # TRANSACTIONS
2135  FOR X = 1 TO TZ
2140  INPUT DE$(X): REM  DATE OF TRANSACTION
2150  INPUT CC$(X): REM CREDIT FLAG =P IF PAYMENT
2160  INPUT CE%(X): REM      INVOICE OR CK#
2170  INPUT AM(X): REM  AMOUNT OF TRANSACTION
2180  INPUT OD(X): REM  AMOUNT OVERDUE
2190  INPUT FC(X): REM  FINANCE CHARGE
2192  INPUT BD(X): REM  BALANCE DUE
2195  NEXT X
2196  INPUT T1$
2198  PRINT D$;"CLOSE"
2199  RETURN
3000  TEXT : HOME
3005  FOR X = 1 TO TZ:DE$(X) = "":CC$(X) = "":CE%(X) =
      0:AM(X) = 0:OD(X) = 0:FC(X) = 0:BD(X) = 0: NEXT
      X:TZ = 0:BE = 0
3010  VTAB 1: PRINT "M>ENU, V>IEW CUSTOMER FILE, OR
      ENTER    CUSTOMER #1 TO ";CN%;: INPUT " TO
      UPDATE ACCOUNT. ";CH$:CH =  VAL (CH$)
3015  IF CH$ = "M" THEN 10
3020  IF CH$ = "V" THEN X = 20: GOSUB 5200: GOTO 3000
3025  IF CH < 1 OR CH > CN% THEN 3000
3027  IF  NOT CF THEN  GOSUB 1300
3028  C% = CH
3030  VTAB 5: PRINT "YOU HAVE SELECTED:": PRINT :
      PRINT "     ";CF$(CH): PRINT : PRINT "IS THIS
      CORRECT (N/CR) ? ";: GET CH$: PRINT CH$: IF CH$
      = "N" THEN 3000
3040  ONERR  GOTO 3043
3041  GOTO 3060
3043  IF  PEEK (222) <  > 5 THEN 30000
3044  POKE 216,0
3045  PRINT "THERE IS NO FILE FOR :": PRINT : PRINT "
      ";CF$(CH): VTAB 23: HTAB 4: PRINT "M> ENU OR T>
      RY ANOTHER # ? ";: GET CH$: PRINT CH$: IF CH$ =
      "M" THEN 10:
3046  IF CH$ = "E" THEN  END
3050  IF CH$ = "U" THEN 3000
3055  GOTO 3000
3060  GOSUB 2100
3100  TEXT : HOME
3110  PRINT "CUSTOMER #";CH;: HTAB 16: PRINT  LEFT$
      (CF$(CH),23)
3115  IF BE < 0 THEN  INVERSE
3120  PRINT "BALANCE $";:N = BE: GOSUB 510: PRINT N$;:
      HTAB 25: NORMAL : PRINT TZ;" ENTRIES"
3130  PRINT "CURRENT TO: ";T1$
3135  PRINT " #    DATE  INV#   DEBIT  CREDIT
      BALANCE";
3140  FOR K = 1 TO 40: PRINT "-";: NEXT : POKE 34,5
3150  FOR K = 1 TO TZ
3152  CV =  PEEK (37): IF CV = 20 THEN  VTAB 23: PRINT
      "PRESS ANY KEY TO CONTINUE ";: GET CH$: HOME
3155  H = 0
3160  N = K: GOSUB 600: HTAB 5: PRINT DE$(K);: HTAB
      13: IF FC(K) THEN  HTAB 14: PRINT "FC";:N =
      FC(K): GOSUB 500: HTAB 24 - NL: PRINT N$:CV = CV
      + 1
3162  HTAB 13: IF CC$(K) = "D" THEN N = CE%(K): GOSUB
      650: PRINT N;
```

contd.

```
3165  IF CC$(K) = "P" THEN  PRINT "PAID";:H = 32: GOTO
      3180
3170  H = 24
3175  IF AM(K) < .01 THEN 3190
3180  N = AM(K): GOSUB 500: HTAB H - NL: PRINT N$;
3190  N = BD(K): GOSUB 500: HTAB 40 - NL: PRINT N$;
3200  NEXT
4000  VTAB 24: CALL  - 958: HTAB 1: PRINT "A>NOTHER,
      M>ENU, P>RINT, OR U>PDATE ? ";: GET CH$: IF CH$
      = "M" THEN 10
4010  IF CH$ =  CHR$ (27) THEN  END
4020  IF CH$ = "M" THEN 10
4030  IF CH$ = "P" THEN 4500
4040  IF CH$ = "A" THEN 3000
4045  IF CH$ <  > "U" THEN 4000
4050  IF CV > 18 THEN  HOME :CV =  PEEK (37)
4060  CV = CV + 2: VTAB CV: CALL  - 958: POKE 34,22:
      GOTO 4100
4100  :
4102  A = BE:: REM  BALANCE
4106  A4 = OD(TZ): REM  OVERDUE
4108  A6 = BD(TZ): REM BALANCE DUE
4109  A8 = BE
4110  TZ = TZ + 1: VTAB CV:H = 0:N = TZ: GOSUB 600
4120  VTAB 23: HTAB 1: PRINT "ENTER DATE OF
      TRANSACTION OR RETURN FOR:     "; MID$ (W$,5,6);:
      VTAB 24: HTAB 24: PRINT "      ";: VTAB 24: HTAB
      24: INPUT "";CH$: IF CH$ = "" THEN A1$ =  MID$
      (W$,5,6): GOTO 4135
4130  A1$ = CH$
4135  VTAB CV: HTAB 5: PRINT A1$
4155  GOSUB 4: PRINT "COMPUTE FINANCE CHARGES (Y/CR) ?
      ";: GET CH$: PRINT CH$: IF CH$ <  > "Y" THEN A5
      = 0: GOTO 4300
4160  VTAB CV: HTAB 14: PRINT "FC"
4175  GOSUB 4: PRINT "ENTER AMOUNT OVERDUE OR RETURN
      FOR:          $";:N = A: GOSUB 500: PRINT N$;:
      INPUT "  $";CH$:A4 =  VAL (CH$): IF CH$ = ""
      THEN A4 = A
4185  GOSUB 4: PRINT "USE 1.5% PER MONTH TO COMPUTE
      FINANCE  CHARGE (N/CR) ? ";: GET CH$: IF CH$ =
      "N" THEN  VTAB 22: HTAB 1: CALL  - 958: VTAB 23:
      INPUT "ENTER INTEREST RATE PER MONTH %";CH$:IR =
      VAL (CH$) / 100: GOTO 4195
4190  IR = .015
4195  A5 =  INT (A4 * IR * 100) / 100
4200  A7 = A5 + A6
4210  N = A5: GOSUB 500: VTAB CV: HTAB 24 - NL: PRINT
      N$
4220  N = A7: GOSUB 500: VTAB CV: HTAB 40 - NL: PRINT
      N$
4225  A8 = A7
4300  GOSUB 4: INPUT "ENTER AMOUNT PAID $";CH$:A3 =
      VAL (CH$): IF A3 THEN A2$ = "P"
4305  VTAB CV + 1: HTAB 13: PRINT "PAID"
4310  N = A3: GOSUB 500: VTAB CV + 1: HTAB 32 - NL:
      PRINT N$
4320  A8 = A8 - A3
4330  IF A8 < 1 THEN A8 =  INT (A8 * 100.5) / 100.5
4340  N = A8: GOSUB 500: VTAB CV + 1: HTAB 40 - NL:
      PRINT N$
4350  VTAB 22: CALL  - 958: VTAB 24: HTAB 7: PRINT "A>
      BORT, C> ORRECT OR S> AVE ";: GET CH$: PRINT
      CH$;: IF CH$ = "A" THEN 4000
4353  IF CH$ = "C" THEN 4360
4355  IF CH$ =  CHR$ (13) OR CH$ = "S" THEN 4400
4357  GOTO 4350
4360  VTAB CV: HTAB 1: CALL  - 958:TZ = TZ - 1:A =
      0:A1$ = "":A2$ = "":A3 = 0:A4 = 0:A5 = 0:A6 =
      0:A7 = 0:A8 = 0: GOTO 4100
4400  BE = A8:DE$(TZ) = A1$:CC$(TZ) = A2$:AM(TZ) =
      A3::OD(TZ) = A4:FC(TZ) = A5:BD(TZ) = A8: GOSUB
      2000
4405  A1$ = "":A2$ = "":A = 0:A3 = 0:A4 = 0:A5 = 0:A6
      = 0:A7 = 0:A8 = 0
4410  GOSUB 4: VTAB 23: PRINT "M> ENU, P> RINT, OR
      UPDATE A> NOTHER ?";: GET CH$: PRINT CH$: IF CH$
      = "M" THEN 5000
4420  IF CH$ = "A" THEN 3000
4430  IF CH$ <  > "P" THEN 4410
4500  TEXT : HOME : VTAB 10: PRINT  TAB( 5);"< OUTPUT
      NOW DIRECTED TO PRINTER >": POKE 34,15
4510  IF  NOT TZ THEN  GOSUB 2100
4515  GOSUB 1100
4520  PRINT D$;"PR#1": PRINT PC$(1);PC$(2):PW = 1:
      GOSUB 10600
4524  PRINT : FOR K = 1 TO 3: POKE 36,59: PRINT
      AD$(K): NEXT
4525  PRINT PC$(4)
4530  PRINT PC$(3): PRINT : FOR X = 1 TO 6: POKE
      36,15: PRINT PC$(4);C$(X): NEXT
4535  PRINT : POKE 36,11: PRINT "STATEMENT OF ACCOUNT
      #";C$;" CURRENT TO ";W$: PRINT : PRINT
4540  PRINT UU$: PRINT : PRINT "TRANSACTION #     DATE
INVOICE #      DEBIT       CREDIT      BALANCE"
4545  PRINT UL$
4550  FOR K = 1 TO TZ
4560  N = K:H = 7: GOSUB 600: POKE 36,16: PRINT
      DE$(K);YR$;: IF FC(K) THEN  POKE 36,33: PRINT
      "FC";:N = FC(K): GOSUB 500: POKE 36,49 - NL:
      PRINT N$
4570  IF CC$(K) = "D" THEN  POKE 36,32:N = CE$(K):
      GOSUB 650: PRINT N;:H = 49: GOTO 4590
4580  IF CC$(K) = "P" THEN H = 62: GOTO 4590
4585  IF CC$(K) <  > "P" THEN 4595
4590  N = AM(K): GOSUB 500: POKE 36,H - NL: PRINT N$;
4595  N = BD(K): GOSUB 500: POKE 36,75 - NL: PRINT N$
4600  NEXT K
4610  PRINT UL$
4620  POKE 36,20: IF BE THEN N = BE: GOSUB 500: PRINT
      : PRINT PC$(4);"  PLEASE REMIT THIS AMOUNT ===>
      $";N$
4630  IF  NOT BE THEN  PRINT : PRINT PC$(4);"THANK YOU
      FOR PAYING PROMPTLY"
4640  PRINT PC$(7)
4700  PRINT D$;"PR#0"
4710  TEXT : HOME : PRINT "RECONCILE ANOTHER (N/CR) ?
      ";: GET CH$: PRINT CH$: IF CH$ = "N" THEN 10
4720  GOTO 3000
4998  PRINT D$;"PR#0"
4999  END
5000  TEXT : HOME : PRINT : HTAB 17: PRINT "TIP (.)":
      HTAB 9
5010  PRINT : HTAB 9: PRINT "> THE INVOICE PRINTER <"
5020  IF  LEN (DA$) < 1 THEN  GOSUB 90
5030  VTAB 6: HTAB 10: PRINT W$;: IF  LEN (W$) < 11
      THEN  PRINT "       19"; RIGHT$ (YR$,2)
5040  V = 11:H = 9
5050  VTAB 8: HTAB 4: PRINT CN$;" CUSTOMERS";: HTAB
      25: PRINT IN$;" INVOICES": VTAB V
5055  VTAB 9: FOR K = 1 TO 40: PRINT "-";: NEXT :
      PRINT
5060  HTAB H: PRINT "<E> XIT": PRINT
5070  HTAB H: PRINT "<I> NPUT DATA": PRINT
5080  HTAB H: PRINT "<P> RINT INVOICE": PRINT
5090  HTAB H: PRINT "<T> OTALS TO DATE": PRINT
5095  HTAB H: PRINT "<U> PDATE/VERIFY ACCT": PRINT
5100  HTAB H: PRINT "<V> IEW CUSTOMER FILE": PRINT
5110  VTAB 23: CALL  - 868: POKE 34,23
5120  HTAB H + 5: PRINT "CHOOSE ONE: ";: GET CH$:
      PRINT CH$
5130  IF CH$ = "I" THEN 8000
5140  IF CH$ = "E" THEN  TEXT : HOME : END
5150  IF CH$ = "V" THEN CH$ = "":X = 23: GOSUB 5190:
      GOTO 5110
5155  IF CH$ = "U" THEN 3000
5160  IF CH$ = "I" THEN 6005
5170  IF CH$ = "P" THEN FL = 1: GOSUB 6000: GOTO 6850
5180  GOTO 5110
5190  IF CN$ < 1 THEN  VTAB 22: HTAB 10: CALL  - 958:
      INVERSE : PRINT "CUSTOMER FILE EMPTY": NORMAL :
      FOR K = 1 TO 1000: NEXT K: RETURN
5200  IF  NOT CF THEN  GOSUB 1300
5210  CF$(0) = "** PRESS SPACE BAR TO END LOOP **"
5220  FOR K = 0 TO CN$
5230  VTAB X: HTAB 1
5240  CALL  - 958: VTAB X: HTAB 5
5250  IF K = 0 THEN  INVERSE : GOTO 5270
```

contd.

```
5260  PRINT K;".  ";
5270  PRINT CF$(K): NORMAL
5280  FOR X1 = 1 TO 750: NEXT X1
5290  IF  PEEK ( - 16384) = 160 THEN K = CN%:GH = 1
5300  NEXT K
5305  IF GH THEN GH = 0: VTAB X: RETURN
5310  GOTO 5220
6000  IF IN% < 1 THEN  VTAB 23: CALL  - 958: INVERSE :
      HTAB 8: PRINT "* NO INVOICES IN MEMORY *":
      NORMAL : FOR K = 1 TO 1500: NEXT K:FL = 0: POP :
      GOTO 10
6005  TEXT : HOME : PRINT "INVOICE #";U$;: HTAB 24:
      PRINT "DATE: ";DA$
6010  VTAB 3: PRINT "SOLD TO:";: CALL  - 868: HTAB 10:
      PRINT US$: PRINT
6015  VTAB 5: HTAB 10: CALL  - 868: PRINT US$
6020  VTAB 7: HTAB 10: CALL  - 868: PRINT US$
6025  VTAB 9: HTAB 10: CALL  - 868: PRINT US$
6030  VTAB 11: PRINT "SHIP TO:";: CALL  - 868: HTAB
      10: PRINT US$: PRINT
6035  VTAB 13: HTAB 10: CALL  - 868: PRINT US$
6040  VTAB 15: HTAB 10: CALL  - 868: PRINT US$
6045  VTAB 17: HTAB 10: CALL  - 868: PRINT US$
6050  VTAB 19: PRINT "SHIP DATE: ";UD$;: HTAB 23:
      PRINT "TERMS: "; LEFT$ (US$,10)
6055  VTAB 21: PRINT "CARRIER: ";US$
6060  VTAB 22: FOR X = 1 TO 40: PRINT "=";: NEXT :
      POKE 34,23
6065  IF FL THEN FL = 0: RETURN
6070  VTAB 23: CALL  - 868: PRINT "<A> <C> <M> <V> OR
      CUSTOMER #1 TO ";CN%;: INPUT ":";CH$
6075  IF CH$ = "A" THEN  GOSUB 800
6076  IF CH$ = "C" THEN  GOSUB 800
6080  IF CH$ = "V" THEN CH$ = "":X = 23: GOSUB 5190
6085  IF CH$ = "M" THEN 5000
6090  C% =  VAL (CH$): IF C% < 1 OR C% > CN% THEN 6070
6095  N = C%: VTAB 1: HTAB 10: GOSUB 650: PRINT
      C%;"-";
6100  IF FL THEN 6110
6105  IN% = IN% + 1:I% = IN%
6110  N = I%: VTAB 1: HTAB 15: GOSUB 650: PRINT I%;
6115  IF FL THEN 6125
6120  PRINT : GOSUB 1100
6125  VTAB 3: HTAB 10: PRINT C$(1)
6130  VTAB 5: HTAB 10: PRINT C$(2)
6135  VTAB 7: HTAB 10: PRINT C$(3)
6140  VTAB 7: HTAB 38: PRINT C$(4)
6145  VTAB 9: HTAB 10: PRINT C$(5)
6150  VTAB 9: HTAB 28: PRINT C$(6)
6155  IF FL THEN 6170
6160  GOSUB 5: PRINT "PRESS <RETURN> IF SAME AS SOLD
      TO.": INPUT "SHIP TO: ";CH$: IF  LEN (CH$) < 1
      THEN  FOR I = 1 TO 6:SH$(I) = C$(I): NEXT I:
      GOTO 6170
6165  GOTO 700
6170  VTAB 11: HTAB 10: PRINT SH$(1)
6175  VTAB 13: HTAB 10: PRINT SH$(2)
6180  VTAB 15: HTAB 10: PRINT SH$(3)
6185  VTAB 15: HTAB 38: PRINT SH$(4)
6195  VTAB 17: HTAB 10: PRINT SH$(5)
6200  VTAB 17: HTAB 28: PRINT SH$(6)
6205  IF FL THEN 6225
6210  GOSUB 5: PRINT "SHIP DATE (";MID$ (DA$,5,6) +
      YR$;"):  _____";: VTAB 23: HTAB 25: INPUT
      "";SD$
6215  IF  LEN (SD$) > 9 THEN  PRINT G$;: GOTO 6210
6220  IF SD$ = "" THEN SD$ =  MID$ (DA$,5,6) + YR$
6225  VTAB 19: HTAB 12: PRINT SD$: IF FL THEN 6245
6230  VTAB 23: CALL  - 958: PRINT "TERMS (CASH) :
      _____";: HTAB 16: INPUT "";TS$
6235  IF TS$ = "" THEN TS$ = "CASH"
6240  IF  LEN (TS$) > 10 THEN  PRINT G$;: GOTO 6230
6245  VTAB 19: HTAB 30: PRINT TS$: IF FL THEN 6260
6250  GOSUB 5: INPUT "CARRIER (UPS) : ";CR$: IF CR$ =
      "" THEN CR$ = "UPS"
6255  IF  LEN (CR$) > 30 THEN  PRINT G$;: GOTO 6250
6260  VTAB 21: HTAB 10: PRINT CR$
6265  GOSUB 6
6270  TEXT : HOME : PRINT "      ITEM            QTY
      PRICE
      SUBTOTAL======================================="
6275  V = 2: POKE 34,23
6280  FOR J = 1 TO 10
6285  IF FL THEN 6295
6290  VTAB 20: CALL  - 958: PRINT "          ITEM
      #";J;: HTAB 19: PRINT ":_____.____.____";:
      VTAB 20: HTAB 20: INPUT "";I$(J): IF  LEN
      (I$(J)) > 20 THEN 6290
6295  IF  LEN (I$(J)) < 1 THEN J = 10: GOTO 6360
6297  IF FL THEN 6330
6300  VTAB 21: CALL  - 868: PRINT "          QUANTITY
      :";: HTAB 20: INPUT "";CH$: IF  VAL (CH$) < 1
      THEN 6300
6305  I(J,0) =  VAL (CH$)
6310  VTAB 22: CALL  - 868: INPUT "   PRICE PER UNIT
      $";CH$: IF  VAL (CH$) < .001 THEN 6310
6315  I(J,1) =  VAL (CH$)
6320  VTAB 23: PRINT "    CORRECT (N/CR) ";: GET CH$:
      PRINT CH$: IF CH$ = "N" THEN 6290
6325  VTAB 23: HTAB 20: PRINT "YES"
6330  I(J,2) = I(J,0) * I(J,1)
6335  IF FL THEN 6345
6340  TL = TL + I(J,2)
6345  VTAB V + J: PRINT I$(J);:N = I(J,0):H = 22:
      GOSUB 600:N = I(J,1): GOSUB 500: POKE 36,30 -
      NL: PRINT N$;:N = I(J,2): GOSUB 500: POKE 36,39
      - NL: PRINT N$
6350  NI = J
6355  CALL  - 958
6360  NEXT J
6365  VTAB V + J + 1: CALL  - 958: POKE 34,23
6370  VTAB V + J: PRINT
      "======================================
            SUBTOTAL  $";
6375  IF FL THEN 6390
6380  IF TL < 0 THEN TL = 0
6385  ST = TL
6390  N = ST: GOSUB 500: POKE 36,39 - NL: PRINT N$
6395  IF FL THEN 6411
6400  GOSUB 5: INPUT " ENTER DISCOUNT %";CH$: IF CH$ =
      "" THEN DC = 0: GOTO 6415
6405  DC = ( VAL (CH$) / 100) * TL
6410  VTAB 15: HTAB 18: PRINT CH$;"%";
6411  N = DC: GOSUB 500: VTAB 15: HTAB 22: PRINT
      "DISCOUNT";: HTAB 40 - NL: PRINT N$
6412  IF FL THEN 6417
6415  CG = TL - DC
6417  N = CG: GOSUB 500: VTAB 16: HTAB 22: PRINT
      "SUBTOTAL";: HTAB 40 - NL: PRINT N$
6420  IF FL THEN 6440
6430  TL = CG
6435  GOSUB 5: INPUT "SHIPPING AND HANDLING FEE
      $";CH$:SF =  VAL (CH$): IF SF < .01 THEN 6445
6440  N = SF: GOSUB 500: VTAB 17: HTAB 9: PRINT
      "SHIPPING AND HANDLING";: HTAB 40 - NL: PRINT N$
6445  IF FL THEN 6460
6450  GOSUB 5: PRINT "STATE TAX AT 5% (N/CR): ";: GET
      CH$: PRINT CH$: IF CH$ <  > "N" THEN TX = CG *
      .05: GOTO 6460
6455  GOSUB 5: INPUT "ENTER STATE TAX %";CH$:CH =  VAL
      (CH$):TX = CG * (CH / 100)
6460  N = TX: GOSUB 500: VTAB 18: HTAB 21: PRINT
      "STATE TAX";: HTAB 40 - NL: PRINT N$
6465  VTAB 19: HTAB 34: PRINT "-------";
6470  IF FL THEN 6500
6475  TL = TL + TX + SF: IF TL < 0 THEN TL = 0
6480  SL = TL
6500  VTAB 22: HTAB 34: PRINT "=======";
6505  IF FL THEN 6515
6510  IF TL < .01 THEN TL = 0
6515  VTAB 23: CALL  - 958:N = TL: GOSUB 500: HTAB 16:
      PRINT "DUE THIS ORDER  $";: HTAB 40 - NL: PRINT
      N$
6520  IF FL THEN  VTAB 20: HTAB 1: PRINT "<M> ENU OR
      ";: VTAB 21: HTAB 1: PRINT "<P> RINT INVOICE ";:
      GET CH$: PRINT CH$: IF CH$ = "M" THEN 10
6525  IF FL AND CH$ = "P" THEN 7000
6530  IF FL THEN  GOTO 6520
```

contd.

```
6535  VTAB 20: HTAB 1: PRINT "<A> BORT OR";: VTAB 21:
      HTAB 1: PRINT "<S> AVE ";: GET CH$: PRINT CH$
6540  IF CH$ = "A" THEN IN% = IN% - 1: GOTO 10
6545  IF CH$ = "!" THEN  END
6550  IF CH$ = "S" THEN B(IN%) = C%: POKE 34,23: VTAB
      20: HTAB 1: PRINT "                ": VTAB 21: HTAB
      1: INVERSE : PRINT "SAVING DATA": NORMAL : GOTO
      6565
6555  VTAB 21: HTAB 8: PRINT " ";
6560  GOTO 6530
6565  FL = 1: GOSUB 1400
6570  PRINT D$;"OPEN INVOICE FILE-";YR$;",L575"
6575  PRINT D$;"WRITE INVOICE FILE-";YR$;",R";I%
6580  PRINT C%: REM    CUSTOMER #
6585  PRINT SD$: REM  SHIP DATE
6590  PRINT ST: REM  ITEM SUBTOTAL
6595  PRINT DC: REM  DISCOUNT
6600  PRINT CG: REM   SUBTOTAL2
6605  PRINT SF: REM   SHIP & HANDLE
6610  PRINT TX: REM  STATE TAX
6625  PRINT TL: REM   TOTAL DUE
6630  PRINT DA$: REM  INVOICE DATE
6635  PRINT TS$: REM  TERMS
6640  PRINT CR$: REM  CARRIER
6645  PRINT NI: REM  # ITEMS
6650  FOR K = 1 TO 6: PRINT SH$(K): NEXT K
6655  FOR K = 1 TO NI: PRINT I$(K): PRINT I(K,0):
      PRINT I(K,1): NEXT K
6660  PRINT D$;"CLOSE"
6662  CH = C%: GOSUB 1900
6665  VTAB 20: HTAB 1: PRINT "<M> ENU OR   ";: VTAB
      21: HTAB 1: PRINT "<P> RINT INVOICE ";: GET CH$:
      PRINT CH$: IF CH$ = "M" THEN 10
6670  IF CH$ = "P" THEN 7000
6675  VTAB 21: HTAB 18: PRINT " ";: GOTO 6665
6680  PRINT D$;"OPEN INVOICE FILE-";YR$;",L575"
6685  PRINT D$;"READ INVOICE FILE-";YR$;",R";I%
6690  IF FL THEN 6485
6695  INPUT C%: REM    CUSTOMER #
6700  INPUT SD$: REM   SHIP DATE
6705  INPUT ST: REM  ITEM SUBTOTAL
6710  INPUT DC: REM  DISCOUNT
6715  INPUT CG: REM  SUBTOTAL2
6720  INPUT SF: REM SHIP & HANDLE
6725  INPUT TX: REM  STATE TAX
6740  INPUT TL: REM   TOTAL DUE
6745  INPUT DA$: REM  INVOICE DATE
6750  INPUT TS$: REM   TERMS
6755  INPUT CR$: REM   CARRIER
6760  INPUT NI: REM   # ITEMS
6765  FOR K = 1 TO 6: INPUT SH$(K): NEXT K
6770  FOR K = 1 TO NI: INPUT I$(K): INPUT I(K,0):
      INPUT I(K,1): NEXT K
6775  PRINT D$;"CLOSE"
6780  RETURN
6850  GOSUB 5: VTAB 23: PRINT "M>ENU OR ENTER INVOICE
      # (1-";IN%;: INPUT ") ";N$:I% =  VAL (N$): IF N$
      = "M" THEN 5000
6855  IF I% < 1 OR I% > IN% THEN 6850
6860  GOSUB 6680: GOSUB 1100
6870  FL = 1: GOSUB 6095
7000  PRINT D$;"PR#1": PRINT PC$(1);
7010  PRINT PC$(2);:PW = 1
7020  GOSUB 10600: PRINT : FOR K = 1 TO 3: POKE 36,59:
      PRINT AD$(K): NEXT K: PRINT
7030  PRINT PC$(3);:PW = 0
7040  PRINT "INVOICE #";:N = C%: GOSUB 650: PRINT
      C%;"-";:N = I%: GOSUB 650: PRINT I%;: POKE
      36,62: PRINT DA$;YR$
7050  PRINT : PRINT "SOLD TO: ";: POKE 36,40: PRINT
      "SHIPPED TO:"
7060  POKE 36,5: PRINT C$(1);: POKE 36,45: PRINT
      SH$(1)
7070  POKE 36,5: PRINT C$(2);: POKE 36,45: PRINT
      SH$(2)
7080  POKE 36,5: PRINT C$(3);: POKE 36,45: PRINT
      SH$(3)
7090  POKE 36,5: PRINT C$(4);: POKE 36,9: PRINT
      C$(5);: POKE 36,45: PRINT SH$(4);: POKE 36,49:
      PRINT SH$(5)
7100  POKE 36,5: PRINT C$(6);: POKE 36,45: PRINT
      SH$(6)
7110  PRINT : PRINT "SHIP DATE: ";SD$;: POKE 36,24:
      PRINT "TERMS: ";TS$;: POKE 36,40: PRINT
      "CARRIER: ";CR$: PRINT : PRINT UU$
7120  POKE 36,13: PRINT "ITEM";: POKE 36,47: PRINT
      "QUANTITY";: POKE 36,62: PRINT "PRICE";: POKE
      36,70: PRINT "SUBTOTAL": PRINT UL$
7130  PRINT : FOR J = 1 TO 10: IF  LEN (I$(J)) < 1
      THEN J = 10: GOTO 7190
7140  IF J < 10 THEN  PRINT " ";
7150  PRINT J;".  ";I$(J);
7160  N = I(J,0):H = 51: GOSUB 600
7170  N = I(J,1): GOSUB 500: POKE 36,66 - NL: PRINT
      N$;
7180  N = I(J,2): GOSUB 500: POKE 36,76 - NL: PRINT N$
7190  NEXT J
7200  NI = J
7210  PRINT UL$: PRINT
7220  POKE 36,55: PRINT "SUBTOTAL  $";:N = ST: GOSUB
      500: POKE 36,76 - NL: PRINT N$
7230  POKE 36,55: PRINT "DISCOUNT";:N = DC: GOSUB 500:
      POKE 36,76 - NL: PRINT N$
7233  POKE 36,70: PRINT "--------"
7235  POKE 36,55: PRINT "SUBTOTAL";:N = CG: GOSUB 500:
      POKE 36,76 - NL: PRINT N$: PRINT
7250  POKE 36,42: PRINT "SHIPPING AND HANDLING";:N =
      SF: GOSUB 500: POKE 36,76 - NL: PRINT N$
7260  POKE 36,54: PRINT "STATE TAX";:N = TX: GOSUB
      500: POKE 36,76 - NL: PRINT N$: POKE 36,70:
      PRINT "--------": PRINT
7290  POKE 36,48: PRINT "DUE THIS ORDER  $";:N = TL:
      GOSUB 500: POKE 36,76 - NL: PRINT N$
7300  PRINT PC$(2): PRINT "*  A FINANCE CHARGE OF 1.5%
      PER MONTH WILL BE ADDED TO ACCOUNTS OVER 30 DAYS
      OLD.": PRINT PC$(3)
7310  HTAB 15: PRINT PC$(4);"THANK YOU FOR YOUR ORDER"
7320  PRINT PC$(3);PC$(6);PC$(7)
7330  PRINT D$;"PR#0"
7340  GOTO 10
8000  IF IN% < 1 THEN  VTAB 23: CALL  - 958: INVERSE :
      HTAB 8: PRINT "* NO INVOICES IN MEMORY *":
      NORMAL : FOR K = 1 TO 1500: NEXT K:FL = 0: GOTO
      5110
8010  TEXT : HOME : VTAB 12: PRINT "THIS OPTION
      REQUIRES A PRINTER IN SLOT 1 THAT IS CAPABLE  OF
      132 COLUMNS. PRESS <RETURN> TO CONTINUE OR <ESC>
      FOR MENU.";: GET CH$: PRINT CH$: IF CH$ =  CHR$
      (27) THEN 5000
8020  IF CH$ =  CHR$ (13) THEN 8040
8030  GOTO 8010
8040  TEXT : HOME : VTAB 12: HTAB 3: PRINT "<OUTPUT
      NOW DIRECTED TO PRINTER>": POKE 34,20: IF B(1) >
      0 THEN 8110
8100  GOSUB 1500
8110  IF  NOT CF THEN  GOSUB 1300
8120  PW = 1: GOSUB 9000:PL = 16: REM  PRINT HEADING
8130  FOR K = 1 TO CN%
8140  GOSUB 9100:PL = PL + 5: REM  CUSTOMER NAME AND
      TITLES
8150  FOR J = 1 TO IN%
8160  IF B(J) = 0 THEN 8270
8170  IF B(J) <  > K THEN 8270
8180  B(J) = 0:PL = PL + 1
8190  PRINT D$;"OPEN INVOICE FILE-";YR$;",L575"
8200  PRINT D$;"READ INVOICE FILE-";YR$;",R";J
8210  INPUT L$: INPUT L$
8220  FOR C = 1 TO 6
8230  INPUT L(C): NEXT C
8240  PRINT D$;"CLOSE"
8250  FOR C = 1 TO 6:LL(C) = LL(C) + L(C): NEXT C
8260  GOSUB 9200:PL = PL + 1: REM  DATA FOR EACH
      INVOICE
8270  NEXT J
8280  GOSUB 9300:PL = PL + 3: REM   SUBTOTALS ALL
      INVOICES FOR  EACH CUSTOMER
8290  FOR C = 1 TO 6:TT(C) = TT(C) + LL(C):LL(C) = 0:
      NEXT C                                          contd.
```

```
8300  NEXT
8310  GOTO 9400: REM  PRINT FINAL TOTALS ALL CUSTOMERS
      AND    ALL INVOICES
9000  PRINT D$;"PR#1": PRINT PC$(1);PC$(2):PW = 1:
      GOSUB 10600
9010  PRINT PC$(3): POKE 36,22: PRINT "TOTALS AS OF
      "W$;YR$: FOR X1 = 1 TO 78: PRINT "*";: NEXT X1:
      PRINT
9020  PRINT D$;"PR#0": RETURN
9100  PRINT D$;"PR#1": PRINT
      PC$(1);PC$(2);PC$(4);"CUSTOMER #";K;"
      ";CF$(K)
9105  PRINT PC$(3)
9110  PRINT : PRINT "INVOICE #";: POKE 36,11: PRINT
      "SHIP DATE";
9112  POKE 36,22: PRINT "ITEM COST";: POKE 36,33:
      PRINT "DISCOUNT";
9115  POKE 36,43: PRINT "SUBTOTAL";: POKE 36,53: PRINT
      "SHIPPING";: POKE 36,63: PRINT "ST TAX";: POKE
      36,71: PRINT "TOTAL DUE";
9117  IF SU THEN  PRINT : GOTO 9122
9120  PRINT : PRINT "=========";: POKE 36,11: PRINT
      "=========";
9122  POKE 36,22: PRINT "=========";: POKE 36,33:
      PRINT "========";
9125  POKE 36,43: PRINT "========";: POKE 36,53: PRINT
      "========";: POKE 36,63: PRINT "======";: POKE
      36,71: PRINT "=========";
9130  PRINT D$;"PR#0": RETURN
9200  PRINT D$;"PR#1": PRINT PC$(1);PC$(3);
9210  N = K: GOSUB 650: PRINT K;"-";:N = J: GOSUB 650:
      PRINT J;: POKE 36,11: PRINT L$;:L = 28
9220  FOR C = 1 TO 6:N = L(C): GOSUB 500: POKE 36,L -
      NL: PRINT N$;:L = L + 10: NEXT C
9230  PRINT D$;"PR#0": RETURN
9300  PRINT D$;"PR#1": PRINT PC$(3);
9305  L = 20
9310  PRINT PC$(1): PRINT PC$(4);"SUBTOTAL";PC$(6);:
      FOR C = 1 TO 6:N = LL(C): GOSUB 500: POKE 36,L -
      NL: PRINT N$;:L = L + 10: NEXT C
9320  PRINT : FOR X = 1 TO 80: PRINT "-";: NEXT X:
      PRINT
9330  IF PL > 55 THEN  PRINT PC$(7):PL = 0
9350  PRINT D$;"PR#0": RETURN
9400  PRINT D$;"PR#1": PRINT PC$(1);: POKE 36,24:
      PRINT PC$(4);"=> T O T A L S <="
9401  FOR X = 1 TO 80: PRINT "*";: NEXT X: PRINT
9402  SU = 1: GOSUB 9112:SU = 0
9406  L = 28
9408  PRINT D$;"PR#1": PRINT PC$(1);
9410  FOR FF = 1 TO 6
9415  N = TT(FF): GOSUB 500
9420  POKE 36,L - NL: PRINT N$;
9425  L = L + 10
9430  NEXT FF
9440  PRINT  CHR$ (12)
9450  PRINT D$;"PR#0"
9460  GOSUB 150
9999  GOTO 10
10000  TEXT : HOME : GOSUB 10500
10010  U$ = "  -  -  ":D$ =  CHR$ (13) +  CHR$ (4)
10020  US$ = "                                        "
10030  UD$ = "          ":YR$ = " 82"
10040  IF C% = 0 THEN C% = 1
10050  DIM I$(10),I(10,4)
10060  G$ =  CHR$ (7)
10070  UZ$ = "    "
10080  UP$ = "  /  -  "
10090  Z1$ = "0":Z2$ = ".0"
10110  REM * YOUR ADDRESS HERE *
10120  AD$(1) = "4524 TUCKERMAN STREET"
10130  AD$(2) = "RIVERDALE, MD.  20737"
10140  AD$(3) = "PHONE    301/###-####"
10150  PC$(1) =  CHR$ (9) + "80N": REM  PRINT 80
       COLUMNS
10160  PC$(2) =  CHR$ (15): REM    COMPRESSED MODE ON
10165  PC$(3) =  CHR$ (18): REM    COMPRESSED MODE OFF
10170  PC$(4) =  CHR$ (14): REM    DOUBLE WIDE ON
10180  PC$(5) =  CHR$ (9) + "132N": REM    PRINT 132
       COLUMNS
10185  PC$(6) =  CHR$ (20): REM    DOUBLE WIDE OFF
10186  PC$(7) =  CHR$ (12): REM    FORM FEED
10190  UU$ =  "**************************************
       ****************************************"
10200 UL$ = "==========================================
      ========================================="
10210  GOTO 11000
10500  TEXT : HOME : SPEED= 200: PRINT
10510  FOR X = 1 TO 40:T$(1) = T$(1) +  CHR$ (126):
       NEXT
10520  T$(9) = T$(1)
10530  T$(2) =  CHR$ (124) + "
                  " +  CHR$ (124)
10540  T$(8) = T$(2)
10550  T$(3) =  CHR$ (124) + "   TTTTTT      IIIIII
       PPPPPP       " +  CHR$ (124)
10560  T$(4) =  CHR$ (124) + "     TT          II
       PP  PP       " +  CHR$ (124)
10570  T$(5) =  CHR$ (124) + "     TT          II
       PPPPPP       " +  CHR$ (124)
10580  T$(6) =  CHR$ (124) + "     TT          II
       PP           " +  CHR$ (124)
10590  T$(7) =  CHR$ (124) + "     TT        IIIIII
       PP      (.)  " +  CHR$ (124)
10600  PRINT : FOR X = 1 TO 9
10620  IF PW THEN  POKE 36,30
10630  PRINT PC$(4);T$(X): NEXT
10640  SPEED= 255
10650  POKE 34,23: RETURN
11000  HOME : VTAB 21: HTAB 19: PRINT "BY JAMES T.
       DEMAY, JR.": GOSUB 90
11010  VTAB 23: PRINT "INSERT DATA DISK AND PRESS ANY
       KEY...";: GET CH$: PRINT CH$: VTAB 23: CALL  -
       958: PRINT "READING DATA FILES...": POKE 34,23:
       GOSUB 1000: GOSUB 1500
11020  DIM DE$(50),CC$(50),CE%(50),AM(50),OD(50),
       FC(50),BD(50)
11030  DIM CF$(CN% + 100)
11035  TQ% = CN%
11040  DIM QQ$(13)
11050  IF IN% > 1 THEN  GOSUB 1500
11060  IF CN% < 1 THEN 808
11070  GOTO 5000
```

NL. All that is required to print the formatted number in a specific coulmn is to HTAB X - NL where X is the column and NL is the length of N$, and then PRINT N$. An alternative method, the one which TIP(.) uses, is to POKE the column desired minus NL into location 36 and then PRINT N$. The latter technique, POKEing location 36, allows printing to columns greater than 40. Refer to the program listing for several examples.

THE PRINTER

I use TIP(.) with an Epson MX-80 printer. By loading the PC$ array with printer control characters in the initialization section of the program, it is easy to control character density, enhanced, double strike, or any of the other print features you may wish to use in your program. The invoices are printed in the 80 column format. Some portions of the Totals to date option are printed using the 132 character mode. TIP(.) could be modified to use a printer capable of less than 132 columns. It may be desirable to eliminate some of the items reported in the Totals section. Be careful if you decide on this course. It may be necessary to rearrange the subroutines which read and write the INVOICE FILEs, in addition to the section that does the actual printing. The Totals section reads and prints the first six items written to each record in the INVOICE FILE. Using this technique simplifies the coding required. A series of FOR-NEXT loops is used to read, total, and print the specific information for each customer.

OTHER MODIFICATIONS

You may decide that some of the data that TIP(.) gathers is not required for your application. Deleting the "ship to" information alone, will result in a savings in the invoice record length of 115 bytes.

Another change that may be required is in lines 90-108. This is the subroutine that gathers the time and date. If you have a THUNDERCLOCK in slot #3, use lines 90 to 97. Lines 105 to 108 can be used if a clock is not available. You may want to substitute a modified subroutine to take advantage of another type of clock in another slot. In either case, keep line 90 since this line is referenced whenever the time and date are needed. A colon or a REM following the line number is all that is required. All other REMs may be deleted with no ill effects to the operation of the program. It is usually poor programming practice to reference REM statements with a GOTO or a GOSUB. REMs are useful tools when used correctly, but they should be included in such a way that their elimination will not affect program operation.

Most certainly, you will want to replace the information in the AD$ array. Put your address and phone number here just as you would like it to appear on your invoices. The T$ array can also be replaced with your company logo. The initialization section, lines 10000 to 10999 contain the original strings. A trial and error approach might be the easiest method for entering data into the T$ array.

REQUIREMENTS

TIP(.) is written entirely in APPLESOFT and will run on a 48K Apple. A single disk drive is required, although a dual drive system is recommended. Using two drives would permit the CUSTOMER FILE to be on one disk and the INVOICE and AR-N-FILEs to be on the other. As previously mentioned, a printer capable of several character sizes is required to work with TIP(.) as written. If other than an EPSON, the printer control characters in the PC string array may have to be altered.

IN SUMMARY

TIP(.) will collect and maintain data on customers and orders generated by them, along with a record of payments. Invoices can be saved, recalled, and printed with a minimum of keystrokes. In addition, a summary of orders to date can be printed. Totals are computed for each customer, and a final total is printed for all customers.

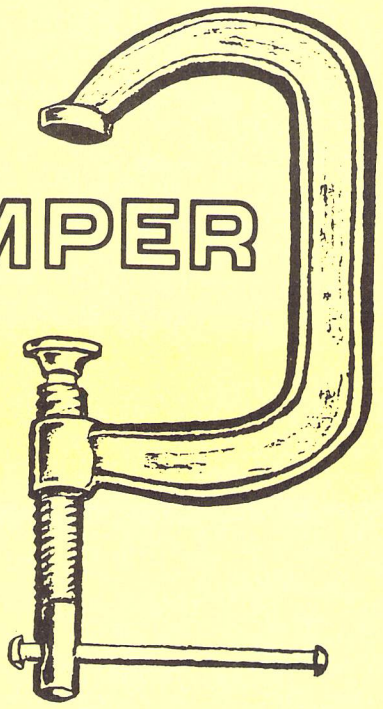
AFTER THOUGHTS

TIP(.) has been running for over six months now with no problems. However, the number of invoices stored per disk is very limited. It may be necessary, depending on the activity of the business, to use a separate data disk each month. I hope that you will be able to adapt TIP(.) to fit your needs. TIP(.) will be available on a WAP library disk in the near future. I am interested in any suggestions and/or improvements you may discover. I can be contacted thru the WAP JOURNAL, or on the WAP ABBS.

```
033D- 60       2300         RTS
033E- 86 1D    2310 NKAR    STX COUN      SAVE X-REG
0340- 18       2320 NEXT    CLC
0341- A5 19    2330         LDA PO1L      INCREMENT POINTER
0343- 69 01    2340         ADC #$01
0345- 85 19    2350         STA PO1L
0347- A5 1A    2360         LDA PO1H
0349- 69 00    2370         ADC #$00
034B- 85 1A    2380         STA PO1H
034D- C5 1C    2390         CMP PO2H      END TABLE?
034F- D0 06    2400         BNE LAAD      NO, LOAD CHAR
0351- A5 19    2410         LDA PO1L
0353- C5 1B    2420         CMP PO2L
0355- F0 0D    2430         BEQ KLAA      YES, FINISHED
0357- A2 00    2440 LAAD    LDX #$00
0359- A1 19    2450         LDA (PO1L,X) LOAD CHARACTER
035B- 49 FF    2460         EOR #$FF
035D- C9 82    2470         CMP #$82      CTRL-B?
035F- F0 DF    2480         BEQ NEXT      YES, SUPPRESS
0361- A6 1D    2490         LDX COUN      RESTORE X-REG
0363- 60       2500         RTS
0364- A5 06    2510 KLAA    LDA $06
0366- 85 36    2520         STA CSWL      RESTORE VECTORS
0368- A5 07    2530         LDA $07
036A- 85 37    2540         STA CSWH
036C- A5 08    2550         LDA $08
036E- 85 38    2560         STA KSWL
0370- A5 09    2570         LDA $09
0372- 85 39    2580         STA KSWH
0374- A6 1D    2590         LDX COUN
0376- A9 98    2600         LDA #$98      CTRL-X
0378- 60       2610         RTS
               2620 *-------------------------------
               2630 *
               2640 * LIST THE CONTENTS OF THE TABLE
               2650 *
               2660 *-------------------------------
0379- A5 AF    2670 XAMN    LDA EOBL
037B- 85 1E    2680         STA PO3L      INIT PTR AT
037D- A5 B0    2690         LDA EOBH      START TABLE
037F- 85 1F    2700         STA PO3H
0381- E6 1F    2710         INC PO3H
0383- A2 00    2720 NCHR    LDX #$00
0385- A1 1E    2730         LDA (PO3L,X) LOAD CHAR FROM TABLE
0387- 49 FF    2740         EOR #$FF      INVERT
0389- 20 F6 FD 2750         JSR COUZ      PRINT IT
038C- 18       2760         CLC
038D- A5 1E    2770         LDA PO3L
038F- 69 01    2780         ADC #$01
0391- 85 1E    2790         STA PO3L
0393- A5 1F    2800         LDA PO3H
0395- 69 00    2810         ADC #$00
0397- 85 1F    2820         STA PO3H
0399- C5 1C    2830         CMP PO2H      ARE WE AT THE
039B- D0 06    2840         BNE STP2      END OF TABLE?
039D- A5 1E    2850         LDA PO3L
039F- C5 1B    2860         CMP PO2L
03A1- F0 14    2870         BEQ ENDE
03A3- AD 00 C0 2880 STP2    LDA $C000     IS A KEY DEPRESSED
03A6- 10 DB    2890         BPL NCHR      NO, GO BY
03A8- 2C 10 C0 2900         BIT $C010     YES, CLEAR STROBE
03AB- AD 00 C0 2910 KEYI    LDA $C000     WATCH FOR NEXT KEYIN
03AE- 10 FB    2920         BPL KEYI
03B0- 2C 10 C0 2930         BIT $C010
03B3- C9 A0    2940         CMP #$A0      =SPACE?
03B5- F0 CC    2950         BEQ NCHR      YES, GO BY
03B7- 60       2960 ENDE    RTS           NO, STOP
               2970         .OR $3C0
03C0- 4C 83 02 2980         JMP INIT      CALL 960-READ CASSETTE
               2990         .OR $3C5
03C5- 4C 79 03 3000         JMP XAMN      CALL 965-LIST THE TABLE
               3010         .OR $3CA      CALL 970-READ TABLE IN BY WAY OF
03CA- 4C 13 03 3020         JMP TABL

0000 ERRORS IN ASSEMBLY
:
```

AMPER CLAMPER



Member Price:

only $19.95

BIG RED APPLE CLUB'S
AMPER CLAMPER

THIS PACKAGE CONTAINS MACHINE LANGUAGE PROGRAMS TO BE USED ON YOUR APPLE][+ OR APPLE //e COMPUTER. THE AMPER CLAMPER ROUTINES ALLOW YOU TO ADD THE POWER AND SPEED OF 6502 MACHINE LANGUAGE TO YOUR APPLESOFT PROGRAMS. THE AMPER CLAMPER PROVIDES A SIMPLE AND FLEXIBLE WAY TO DO THIS WITHOUT REQUIRING YOU TO HAVE ANY KNOWLEDGE OF MACHINE LANGUAGE. USING AMPER CLAMPER, YOU CAN SIMPLY CLAMP THE MACHINE LANGUAGE ROUTINES RIGHT ON TO YOUR APPLESOFT PROGRAMS AND GIVE EACH ROUTINE A NAME THAT IS EASY FOR YOU TO REMEMBER. YOU DON'T HAVE TO WORRY ABOUT WHERE AND WHEN THEY ARE LOADED. IT IS ALL DONE FOR YOU BY AMPER CLAMPER.

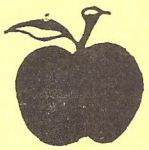
AMPER CLAMPER COMES PREPACKAGED WITH OVER TWENTY MACHINE LANGUAGE ROUTINES THAT YOU CAN START USING IN YOUR PROGRAMS RIGHT AWAY. SOME OF THESE INCLUDE WILDCARD ARRAY SEARCH, FAST TEXT FILE READ AND WRITE, COMPUTED GOSUB AND GOTO, FAST ARRAY SORT, A DYNAMIC ARRAY ROUTINE THAT ALLOWS YOU TO INSERT ITEMS INTO THE MIDDLE OF AN ARRAY, AND MANY MORE. AMPER CLAMPER ALSO ALLOWS YOU TO ADD ROUTINES THAT YOU HAVE WRITTEN OR TYPED IN.

AMPER CLAMPER COMES TO YOU ON AN UNPROTECTED STANDARD DOS 3.3 DISKETTE.

THE BIG RED APPLE CLUB

THE BIG RED APPLE CLUB
1301 N. 19th
Norfolk, NE 68701
402-379-3531

BULK RATE
U.S.Postage Paid
Bulk Permit 65
Norfolk, NE



FORWARDING & RETURN POSTAGE GUARANTEED

APPLICATION FOR MEMBERSHIP

BIG RED APPLE CLUB, 1301 N. 19th, Norfolk, NE 68701

MEMBERSHIP FOR:

NAME: _____

ADDRESS: _____

CITY: _____ STATE_____ ZIP_____

☐ $12.00 payment enclosed. Add $16.00 for overseas airmail postage. Please make payment in U.S. dollars

☐ MASTERCARD ☐ VISA

Card Number |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_| Expiration Date _____

Your Signature _____ Phone___ ___